ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
Y DE TELECOMUNICACIÓN

Titulación :

INGENIERO EN INFORMÁTICA

Título del proyecto:

DESARROLLO DE APLICACIÓN IPHONE PARA LA CREACIÓN Y COMPARTICIÓN DE FOTOS CREATIVAS

Alumno: Oleh Kudinov

Tutor: Oscar Ardaiz Villanueva

Pamplona, 25 de Julio de 2012

Agradecimientos

Estos seis años de carrera han sido muy especiales y también duros. Se ha puesto a prueba tanto mi capacidad de aprender como mi capacidad de crecer como persona. Al final de esta primera etapa de mis estudios quería dedicar unas palabras de agradecimiento a las personas que tanto me han ayudado:

- A mis padres, porque sin ellos no sería quien soy ni tendría lo que tengo.
- Al Dr. Oscar Ardaiz, tutor de este proyecto, que me brindó una gran oportunidad al mismo tiempo que me otorgó la libertad de trabajar en algo que realmente me gusta.
- A mis amigos Alberto, Alex, Asier, etc. por demostrarme que la verdadera amistad existe.
- A mis colegas de clase, buenos compañeros y excelentes amigos.
- Al área de Empleo de la Fundación Universidad Sociedad de UPNA , que con una oferta de trabajo me dio la motivación necesaria para terminar este proyecto y empezar una nueva etapa de mi vida.

A todos ellos, gracias.

1. INTRODUCCIÓN

La creatividad es muy importante en las sociedades modernas, ya que es la habilidad básica que conduce a la solución de problemas, al igual que la innovación, ambas muy demandadas en la mayoría de las organizaciones económicas y sociales. La creatividad, denominada por algunos autores como el pensamiento divergente, es la capacidad de los seres humanos para producir algo nuevo y útil, y para resolver los problemas de una forma original.

La creatividad puede ser ejercitada y mejorada a través de diversas técnicas. La técnica de lluvia de ideas se basa en el hecho de que los grupos sociales pueden estimular la creatividad, sobre todo si un grupo está formado por un conjunto heterogéneo de personas con diferentes orígenes.

La creatividad puede ser puesta en práctica utilizando herramientas informáticas que facilitan los procesos de grupo o por medio de interfaces de ordenador avanzadas [ver referencia 2]. Hemos diseñado un juego con el fin de desarrollar la creatividad y la capacidad de innovación, aprovechando las características exclusivas de los teléfonos inteligentes: una cámara fotográfica, la comunicación inalámbrica de red y una capacidad de procesamiento considerable. Para este objetivo hemos diseñado un juego social que hace uso de una técnica de combinación de imágenes para fomentar la creatividad.

Numerosos juegos para crear nuevas ideas han sido diseñados y probados en[ver referencia 1]. Sus juegos basados en tarjetas que se jugarán presencialmente por un grupo facilitó la aparición de nuevos conceptos de juego. La técnica combinada ha sido utilizada para crear nuevos diseños con muchos de los participantes utilizando una plataforma en línea para el crowdsourcing produciendo resultados muy creativos [ver referencia 3]. Es obvio que hay varios juegos para teléfonos inteligentes que utilizan fotografías como elementos de juego (Appysnap, ARIS), sin embargo la creatividad no es su objetivo.

Optamos por hacer un juego cuyo fin fuese la creatividad, sobre la base de la toma de fotografías por el hecho de que la fotografía es una de las actividades más creativas. Por otra parte requiere pocos conocimientos técnicos y pocos recursos, a parte de la cámara. Las cámaras digitales han aprovechado esta técnica porque la toma de una imagen no tiene ningún costo y las cámaras tienen gran capacidad de almacenamiento. Además, la conectividad a Internet de los teléfonos móviles permite la creación de juegos sociales en los que los resultados creativos de un usuario pueden inspirar a otros usuarios para crear ideas diferentes. Además los usuarios pueden evaluar las creaciones de otros usuarios, facilitando la formación de un ranking. Otra ventaja importante de la conectividad a Internet es la posibilidad de incluir un mecanismo para que el juego pueda auto-generar sus contenidos con las creaciones de los demás participantes, lo que permite un número infinito de nuevos niveles.

El proyecto consistirá en el desarrollo de una aplicación que ofrece a los usuarios las

funciones para la creación y compartición de imágenes creativas de forma colaborativa. Con esta aplicación se podrá sacar fotos, subirlas y participar como si fuera una “red social” fotográfica. El juego se basará en la técnica de creatividad denominada "combinación": las nuevas ideas son el resultado de la combinación o asociación de ideas previas. El juego exigirá al jugador crear nuevas fotografías que combinen dos imágenes existentes. Los jugadores deberán crear una película que combina dos imágenes al azar, lo que requerirá el ejercicio de sus habilidades creativas. Los jugadores podrán optar por un par aleatorio diferente, sin embargo, se perderán algunos puntos de su puntuación como penalización. Las imágenes que se emplearan en las combinaciones serán inicialmente pre-configurables por los desarrolladores, pero estas últimas mejores fotografías seleccionadas por los usuarios serán elegidas para estar en un nuevo conjunto de imágenes que serán combinadas por otros usuarios. De esta manera el juego generará automáticamente su contenido para permitir el funcionamiento autónomo.

2. ANTECEDENTES Y OBJETIVOS

2.1 Antecedentes

Para la realización de este proyecto se analizaron muchas aplicaciones existentes en la actualidad similares a la aplicación que se debe desarrollar para saber como funcionan y que ofrecen. Después se escogió una para hacer un estudio más detallado y sacar de ella los conceptos básicos. Estos se describen a continuación.

2.1.1 Piictu

Piictu es una de las ideas fotográficas más originales de cuantas hayamos visto en el blog *appleblog.com* durante el año, salvo la honrosa excepción de *Wall* y alguna parecida como *Color*. Aunque el concepto difiere enormemente del resto de las que hemos comentado. En Piictu la imagen no es un objeto artístico, es parte de la conversación. Por eso en Piictu no verás efectos, no verás filtros, no verás imágenes estilo años 70 o con aroma a Polaroid, solo verás fotos sencillas, sin retoques.

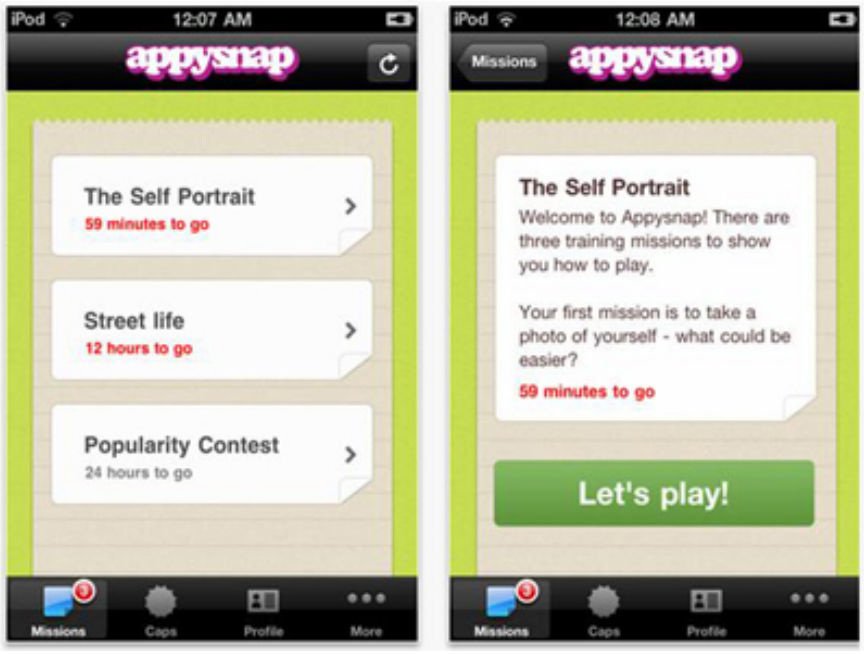
Pero entonces, ¿qué es Piictu?. Piictu es una aplicación que funciona como una red social, en este sentido se parece a Instagram, pero ahí acaban sus similitudes. Para entenderlo deberíamos remontarnos a la idea que inspiró su concepción. La idea surgió mientras un grupo de amigos veían el Mundial desde sus casas y cada uno enviaba a través de MMS su foto con las camisetas de sus equipos de futbol. Allí notaron que la mayoría de las imágenes no serían vistas de nuevo pero, a pesar de eso, crearon un entorno de enorme diversión. Esto los llevó a identificar una nueva tendencia: las imágenes como objetos de interacción.

En Piictu cada *stream*, cada conversación, es una línea, una fila de imágenes que va creciendo a medida que los pertenecientes a la comunidad añaden sus aportaciones. Imagina que estás tomando algo y creas un nuevo stream con la foto de la bebida que tienes en las manos, quizás otros usuarios cuelguen las suyas en idénticas circunstancias… ya se creó una experiencia interesante.
Nada más instalarla, veremos tres opciones arriba, la primera llamada *Following* muestra lo que comparten las personas a las que seguimos (personas que podemos añadir desde nuetsra red de contactos en Twitter o Facebook), la segunda *Popular* los streams con más participantes y *Latest* lo último que se ha subido a la red. En la parte baja además podremos consultar nuestra actividad, modificar nuestro perfil, o buscar nuevos individuos a los que seguir.

Nuestras capturas pueden añadirse a Piictu de dos formas. Por un lado pulsando sobre el icono de cámara abajo en el centro, con lo que abriremos una nueva línea de fotografías o *stream*. O bien podemos pulsar sobre el último recuadro de los *streams* ya empezados para añadir nuestra aportación, que en la empresa Piictu han llamado Piics (una fotografía usada como respuesta). Una vez realizada, siempre cabrá la posibilidad de compartir la imagen en Facebook y Twitter.

Realmente es una app muy original, que al contrario de lo que le sucede a otras redes sociales creadas por y para iOS, ya posee una lista considerable de usuarios, de modo que no será problema encontrar momentos para participar. Además pronto también estará disponible en Android para aumentar aún más las posibilidades de interacción con nuestros amigos.

2.1.2 Appysnap



Appysnap es una app para combinar y jugar con fotos. En este juego hay que completar las "misiones" de fotos para ganar puntos y premios de Appysnap. Las misiones se completan sacando fotos con la cámara del teléfono.

Cuando empiezas a jugar se tienen tres misiones de entrenamiento para aprender. La primera misión es tomar una foto de sí mismo.

Actualmente la aplicación no está disponible. El equipo de desarrollo de la aplicación no ha conseguido la financiación suficiente para pagar todos los gastos de servidores, diseño, promociones y premios. Para crear esta aplicación los desarrolladores estuvieron trabajando durante 15 meses sin recibir remuneración alguna.

2.2 Objetivos

El objetivo de este proyecto es construir una aplicación para IOS, la cual permita a los usuarios realizar fotos creativas y compartirlas con el resto del mundo empleando un Smartphone. Las fotos se tomarán utilizando la cámara de que dispone el IPhone. Además la aplicación permitirá a los usuarios elegir y recortar los fotos de los álbumes de sus IPhone. Una vez elegidas y recortadas se subirán automáticamente al servidor y todos los demás usuarios podrán ver estas imágenes.

En principio los usuarios podrán crear fotos originales combinando fotos de otros usuarios y ganar puntos si sus fotos son elegidas como mejores. La foto con mayor número de puntos para una combinación la llamaremos Picthru. Las mejores fotos Picthru se coinvertirán en las fotos con que se formarán nuevas combinaciones.

La foto que hará el usuario tendrá que tener alguna relación o significado con la combinación de fotos elegida. Por ejemplo, si la combinación elegida es una foto de agua y otra foto de fuego entonces una foto relacionada que podría hacer el usuario podría ser de vapor o de una bebida.

El administrador tendrá que ser capaz de consultar y elegir las mejores fotos a través de la pagina web. Inmediatamente después de la elección de las mejores fotos, los usuarios podrán formar nuevas combinaciones con estas fotos en sus IPhone.

En la página web de la aplicación, también podrá verse el árbol completo de combinaciones a partir de las cuales se haya creado cualquier imagen. En el IPhone , sin embargo, solo se podrá visualizar una parte de este árbol debido a las limitadas dimensiones del dispositivo.

La aplicación dispondrá de una pantalla con el ranking de los usuarios. En esta pantalla se podrá ver una lista de jugadores, ordenada en función del número de puntos que tiene cada uno. De esta manera los jugadores podrán competir con sus amigos para demostrar su creatividad. Además los usuarios podrán seguir a los mejores jugadores para ver sus fotos y adjudicar sus votos a las fotos que prefieran. Las fotos con mayor número de votos se convertirán en Picthrus.

Para hacer la aplicación más atractiva se añadirá la funcionalidad adicional de recibir notificaciones de actividad dentro de la aplicación. El usuario recibirá las notificaciones en el caso de que a alguien le guste su foto o de que un jugador empiece a seguirle.

A día de hoy la mayoría de los aplicaciones están integrados con las redes sociales como Facebook y Twitter. Las ventajas de las redes sociales son muy numerosas. Su atractivo esencial radica en la participación e interacción que proporcionan a los internautas. La relación entre los usuarios en estas plataformas pasa de ser vertical a horizontal, posibilitando que todos estén al mismo nivel. Cualquiera puede convertirse en emisor y producir sus propios contenidos.

Por lo tanto se añadirán funcionalidades que permitan a los usuarios de la aplicación a subir las fotos desde IPhone directamente a Facebook. Esto permitirá incrementar la promoción de nuestra aplicación, haciendo que sea conocida por muchas mas personas.

3. DESARROLLO DEL TRABAJO

Para llevar a cabo nuestro proyecto, el proceso de desarrollo se ha dividido en las etapas de análisis, diseño, codificación y pruebas de la aplicación. Así pues se explicaran los diferentes pasos realizados para cumplir los objetivos predeterminados:

- Etapa 1: Análisis de los tipos de software de caracter social existentes en la actualidad, y una vez que se tenga una idea general de lo que existe actualmente, se procederá a la especificación de requisitos de la aplicación y de las herramientas destinadas al diseño e implementación de la misma.
- Etapa 2: Diseño de la aplicación. En esta etapa se creará el modelo entidad-relación, diagrama de clases, diagramas de caso de uso y de actividad por cada caso de uso. De esta manera quedarán establecidos los pilares para poder pasar a la siguiente fase.
- Etapa 3: Implementación. A partir del diseño se programará la aplicación empleando las herramientas especificadas y durante esta fase se irán realizando pequeñas pruebas para comprobar su correcto funcionamiento.
- Etapa 4: Pruebas. El objetivo es corregir posibles fallos o funcionamientos incorrectos de la aplicación. Se pueden desglosar en pruebas unitarias (caja blanca y caja negra), de integración y de aceptación.
- Etapa 5: Documentación. Elaboración de los diferentes apartados de la memoria no elaborados en las fases anteriores y refinamiento de lo ya elaborado.

La estructura del proyecto está compuesta por las siguientes partes:

a) Análisis y comprensión de los requisitos del sistema
b) Evaluación de un software existente similar al nuestro
c) Desarrollo de un prototipo de la interfaz gráfica e implementación de la aplicación.

Se analizarán todos los requisitos que debe cumplir la aplicación (tiempos de descarga y la presentación de las imágenes, subsistema de gestión de puntuaciones, notificaciones, tratamiento de errores debidos a perdida de conexión a Internet, etc…).

En la segunda parte, se estudiará uno de los campos más florecientes de iOS; las aplicaciones fotográficas. Un buen ejemplo a considerar es Piictu. Piictu es una aplicación móvil de conversación y juegos con fotos. Sin necesidad de escribir**,** los usuarios se pueden comunicar sacando fotos desde su móvil y recibiendo otras fotos como respuesta, generando un contenido visual muy atractivo. Se deberá estudiar la forma de mejorar la interfaz gráfica de dicho software y cómo adaptar la funcionalidad de nuestra aplicación.

La última parte corresponderá con el desarrollo del diseño y la implementación de la aplicación.

Este sistema resultante será muy valioso para estimular la creatividad de los usuarios.

3.1 PICTHRU

3.1.1 ANÁLISIS

Este paso se divide en dos pasos: análisis de los requisitos y análisis del sistema.

3.1.1.1. ANÁLISIS DE LOS REQUISITOS

El objetivo de este paso es establecer una base teórica muy clara para no tener que decidir los aspectos importantes durante el diseño. Eso significa que hay que obtener los requisitos de sistema y buscar la información importante para el proyecto. Como los dos apartados son dependientes entre si, los dos pasos se realizan uno detrás de otro.

Análisis y especificación de requisitos

En este paso tenemos que conseguir los datos explicados de punto 2.1, Objetivos de este documento. La información es obtenida en las reuniones con el tutor y otros medios como el análisis de las aplicaciones similares. El objetivo de proyecto fue determinado cuando el alumno y tutor llegaron a un consenso.
Es muy importante establecer todos los requisitos y que estos no cambien. En el caso de que los requisitos estén mal especificados, esto nos puede llevar a finalización de proyecto sin haber conseguido los objetivos, por este motivo este paso tiene gran importancia.

Búsqueda de información

El objetivo es buscar información que nos pueda servir de ayuda durante todo el desarrollo de aplicación. Se analizaron distintos enlaces Web, los cuales se especifican en bibliografía de este documento. La información fue buscada de la siguiente manera:

1. Buscar aplicaciones o sistemas similares.
2. Buscar tecnologías que pudiesen ser utilizadas.
3. Buscar manuales sobre lenguaje de programación.

3.1.1.2. ANÁLISIS DEL SISTEMA

Casos de uso

Para hacer los casos de uso de sistema hay que primero determinar los actores y averiguar los requisitos. En principio tenemos claro que hay un administrador de nuestra aplicación. También existen usuarios registrados del sistema, que pueden realizar prácticamente todas las funciones disponibles. También hay usuarios no registrados, que tienen solo algunas funciones disponibles. En resumen tendremos siguientes autores: usuario no registrado, usuario registrado y administrador.

Las funcionalidades requeridas por el sistema pueden dividirse en tres grupos: funciones básicas de visualización de fotos, funciones de uso de aplicación y funciones especiales de administrador. Así pues, el diagrama resultante es la figura 1.

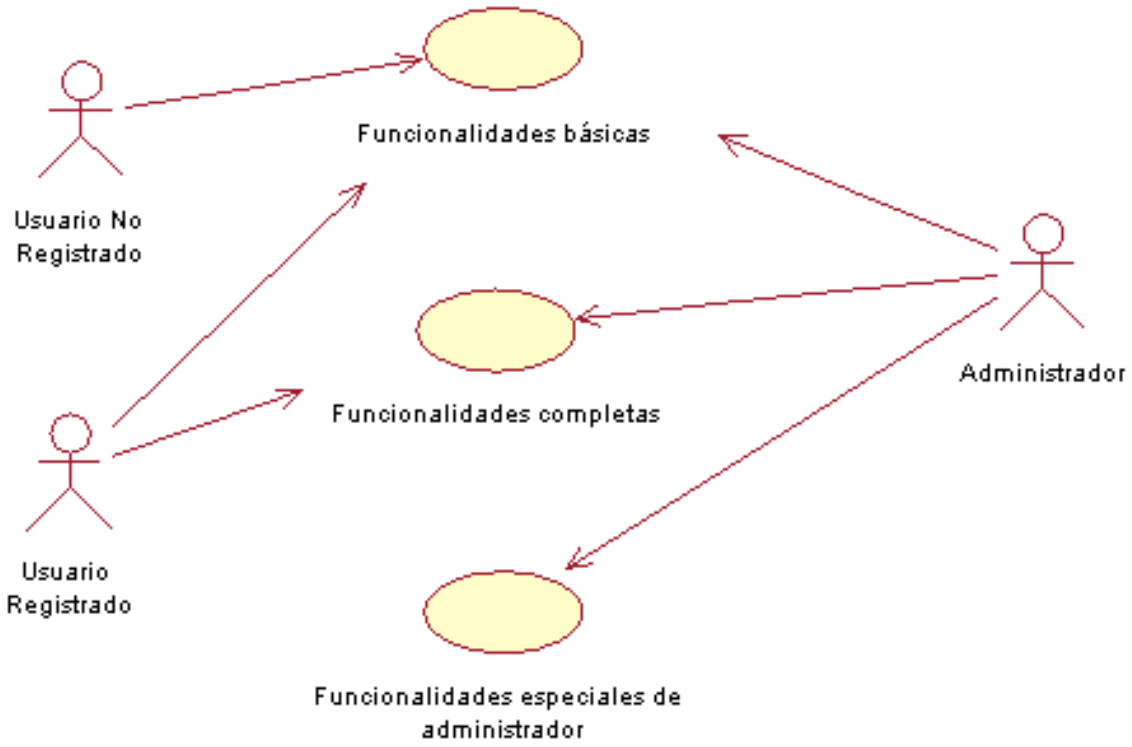


Figura 1

Ahora se realizará la descomposición de cada caso de uso. Se describirán los casos de uso para cada actor.

Casos de uso de las funcionalidades básicas

Registro: Cada usuario podrá crear una cuenta en el sistema. Para registrarse el usuario deberá introducir los datos como el nombre de usuario y la contraseña. Si el usuario existe o la contraseña está mal el sistema le avisara y el usuario podrá intentarlo de nuevo hasta conseguir un registro valido.

Visualizar fotos: Los usuarios no registrados podrán ver todos los fotos hechas por los usuarios registrados.

Visualizar un conjunto de picthrus(fotos a combinar): Los usuarios no registrados podrán ver solo un conjunto de las fotos con las que se hacen los combinaciones, estas fotos se llamas picthrus.

Consultar ayuda: Todos los usuarios podrán acceder a la pantalla de ayuda.

Así pues, el diagrama resultante es la figura 2.

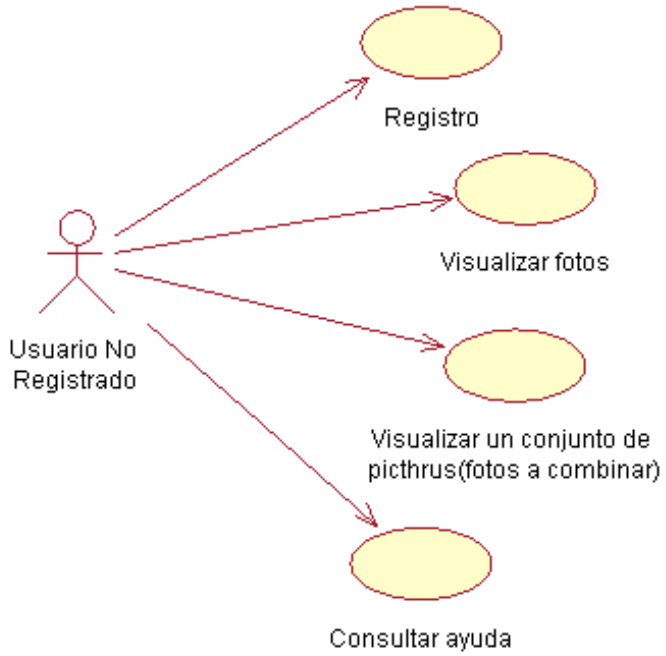


Figura 2

Casos de uso de funciones especiales de administrador

En nuestro sistema el administrador podrá administrar los usuarios y las fotos. Esto sirve en los casos en que las fotos subidas son de contenido inadecuado, para poder eliminarlas.
Tenemos los siguientes casos de uso:

Escoger fotos para un nivel: El administrador podrá escoger a través de un sitio web, las fotos para combinar en el siguiente nivel.

Crear niveles: Después de haber elegido fotos para combinar, el administrador creará el siguiente nivel.

Eliminar fotos: El borrado consiste en una sustitución del campo "dirección url" de la foto por otra dirección de una imagen en blanco o una imagen inexistente. El borrado de fotos se realizara con el gestor de la base de datos phpMyAdmin.

Eliminar usuarios: El borrado de usuarios se realizara igualmente con el gestor de la base de datos phpMyAdmin.

Así pues, el diagrama resultante es la figura 3

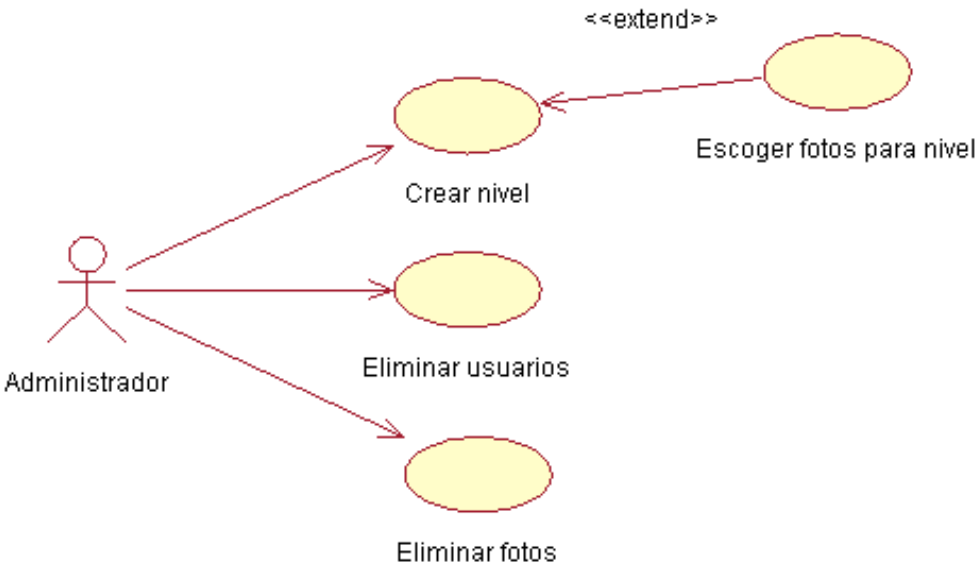


Figura 3

Casos de uso de las funcionalidades completas

Los actores usuario registrado y el administrador tendrán los siguientes casos de uso:

Login: El login es la autentificación en el momento de ingresar al sistema. Para loguearse se le pedirá el nombre y la contraseña. Si la cuenta con este nombre existe y la contraseña está bien, entonces pasará a ser usuario registrado.

Logout: El usuario registrado tendrá la opción de cerrar su sesión en el sistema. Para esto en la pantalla de su perfil tendrá un botón con esa opción.

Combinar fotos: Los usuarios podrán formar nuevas combinaciones con las mejores fotos Picthrus en sus IPhone.

Elegir el conjunto de picthrus(fotos a combinar): El usuario será capaz de cambiar el conjunto de fotos a combinar.

Subir fotos al servidor: El usuario podrá subir cualquier foto relacionada con una combinación desde su iPhone.

Subir fotos a Facebook: Este caso de uso será necesario para subir las fotos desde IPhone directamente a Facebook.

Hacer fotos con cámara: El usuario será capaz de hacer una foto con la cámara de su iPhone.

Elegir fotos de álbum: Como segunda alternativa a hacer fotos con la cámara, el usuario podrá seleccionar cualquier foto desde sus álbumes que están en el iPhone.

Recortar fotos: Con este caso de uso podremos recortar cualquier área cuadrada de la foto seleccionada del álbum.

Ver el árbol entero de origen de una foto: En la página web de la aplicación, también podrá verse el árbol completo de combinaciones a partir de las cuales se haya creado cualquier imagen. En el IPhone , sin embargo, solo se podrá visualizar una parte de este árbol debido a las limitadas dimensiones del dispositivo.

Ver fotos de usuarios: El usuario será capaz de ver todas las fotos que han hecho los usuarios.

Ver fotos mías: El usuario podrá ver fotos en grande que fueron hechas por el.

Ver fotos de usuarios que sigo: El usuario podrá ver fotos de los usuarios a los que está siguiendo.

Aumentar fotos: Como las imágenes de cada categoría van a ser de tamaño pequeño, tendremos que agrandarlas cada vez que el usuario pulsa sobre ellas. También, el usuario podrá ver la información de la foto que esta viendo.

Seguir a los usuarios: Además los usuarios podrán seguir a los mejores jugadores para ver sus fotos y adjudicar sus votos a las fotos que prefieran.

Ver a los usuarios que me siguen y a los que sigo yo: El usuario podrá ver a todos los usuarios que le están siguiendo y a todos los usuarios que está siguiendo él.

Votar por fotos: Los usuarios podrán adjudicar sus votos a las fotos que prefieran. Las fotos con mayor número de votos se convertirán en Picthrus.

Enviar y Recibir notificaciones: Los usuarios recibirán notificaciones de actividad dentro de la aplicación. El usuario recibirá las notificaciones en el caso de que a alguien le guste su foto o de que un jugador empiece a seguirle.

Ver la actividad de los usuarios: El usuario podrá ver la hora de las actividades realizadas por los otros usuarios.

Ver la historia de obtención de puntos: El usuario podrá ver como y cuando se obtenido sus puntos.

Ver el historial de actividad de los usuarios: El usuario será capaz de ver la historia de actividades de los usuarios.

Ver el ranking de usuarios: La aplicación dispondrá de una pantalla con el ranking de los usuarios. En esta pantalla se podrá ver una lista de jugadores, ordenada en función del número de puntos que tiene cada uno.

El diagrama de casos de uso es la figura 4.

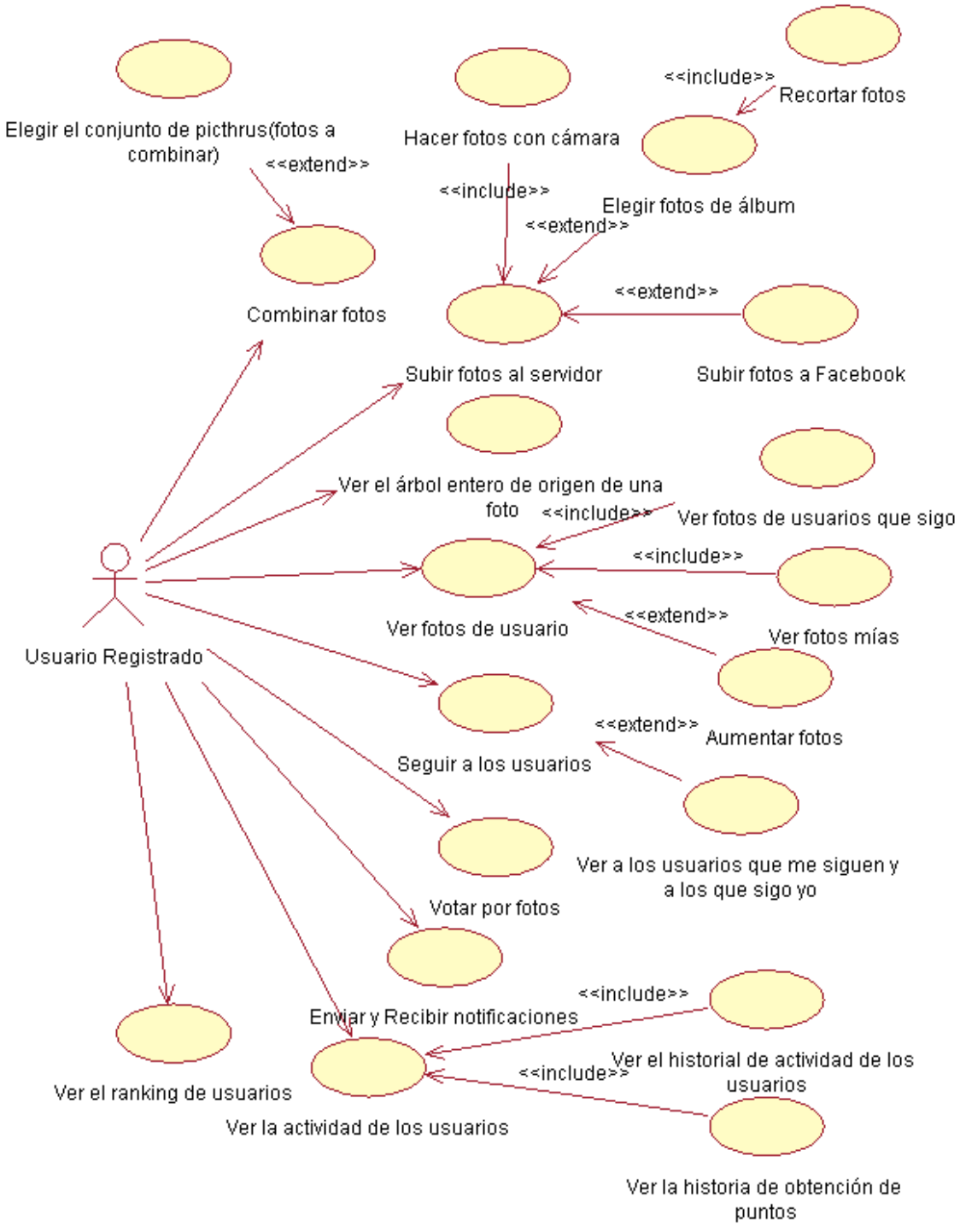


Figura 4

Arquitectura del sistema

En este apartado se describirá la organización fundamental de este sistema, que incluye sus componentes, las relaciones entre sí y el ambiente, y los principios que gobiernan su diseño y evolución.

La aplicación Picthru va a correr en los dispositivos iPhone de los usuarios. Para transferir las fotos y los datos a servidor de Pichtru la aplicación utilizará Restkit con una conexión HTTP Auth. Los componentes APNS servidor y APNS proveedor son utilizadas para el envío de notificaciones. También vamos a tener un servidor de backup, al cual vamos transferir las copias de seguridad mediante la conexión segura SSL.

Así pues, el diagrama resultante es la figura 5.

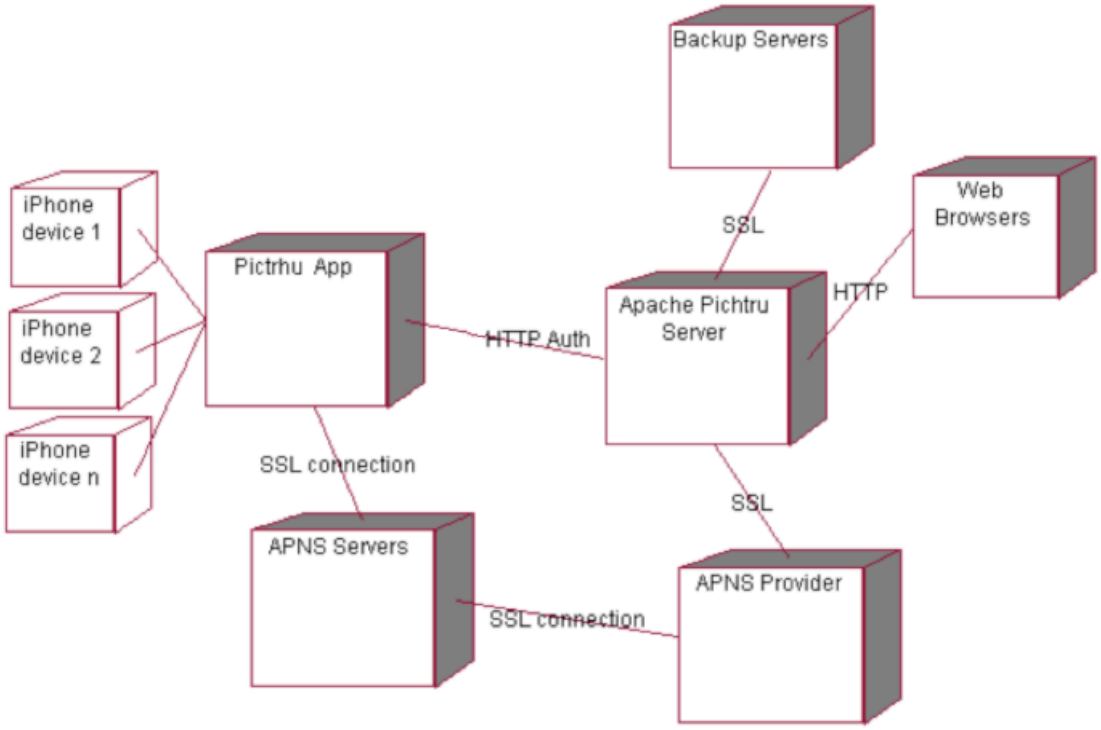


Figura 5

Existen muchos patrones de diseño para organizar el código y la forma de programar aplicaciones. En la programación para iOS y CakePHP utilizaremos el MVC o Model View Controller (Modelo-Vista-Controlador)**.**

El MVC se explica mejor con la figura 6:

- El usuario interactúa con la aplicación y la vista (U objeto de la vista) a través de la interfaz de usuario.
- La vista le envía un mensaje al controlador diciéndole por ejemplo que hemos presionado un botón y queremos que esto responda a alguna acción.
- El controlador recibe el mensaje y contacta con el modelo para realizar la acción y actualizar la información pertinente.

- El controlador recoge la información requerida por la vista pero actualizada por el modelo y por ultimo actualiza la vista con los cambios que se han hecho en el modelo.

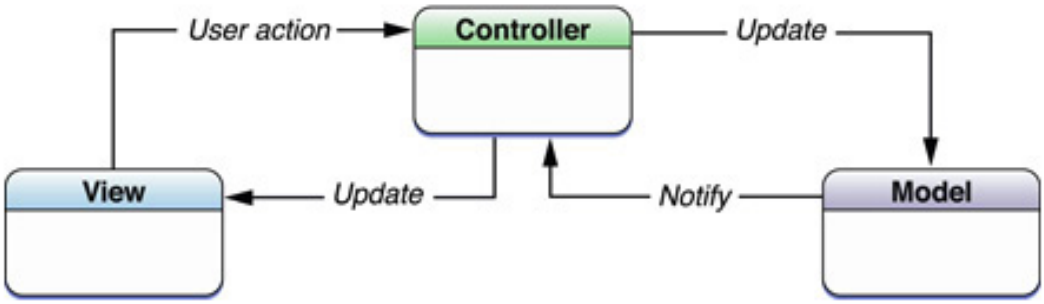


Figura 6

En la parte del cliente programada con Xcode vamos a tener el modelo de datos formado por las entidades que se puede ver en la ilustración 1.

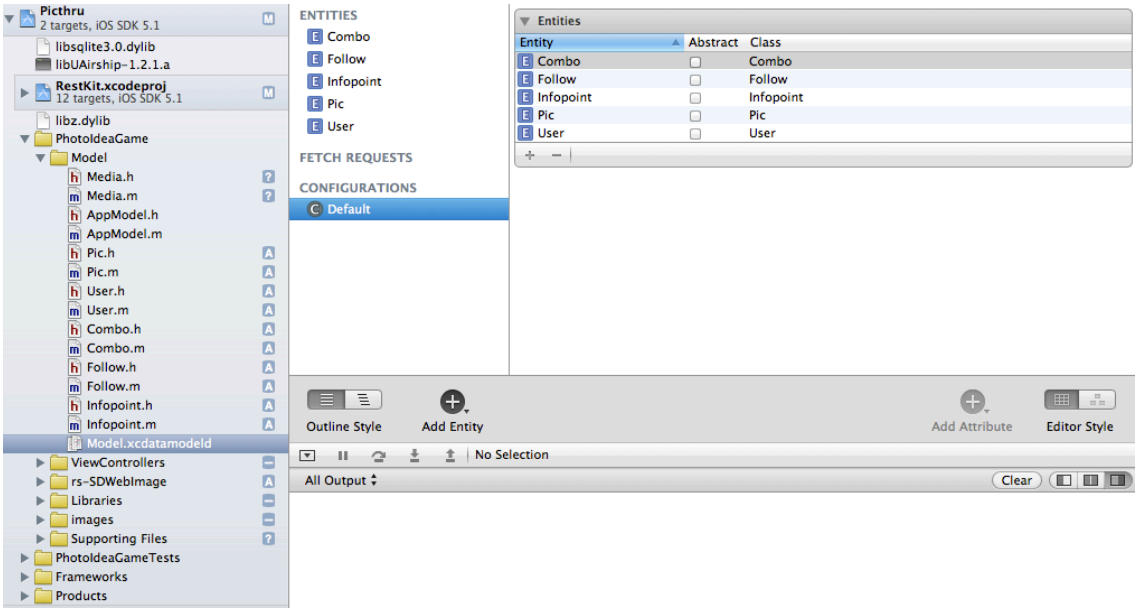


Ilustración 1

El modelo de datos de este sistema en la parte del cliente se explica mas detalladamente en el apartado de Diagrama Entidad-Relación. En el apartado paso a tablas del diagrama de relación se describe la Base de datos resultante en la parte de servidor. Por lo tanto es el modelo de datos en la parte del servidor.

Diagrama de clases

Como la aplicación estará implementada con Objective C, php y html, solo podemos realizar el diagrama con Objective C porque en php no realizaremos objetos. Las funciones y atributos que tienen menor importancia se han omitido para mayor claridad.

El diagrama resultante es la figura 7.

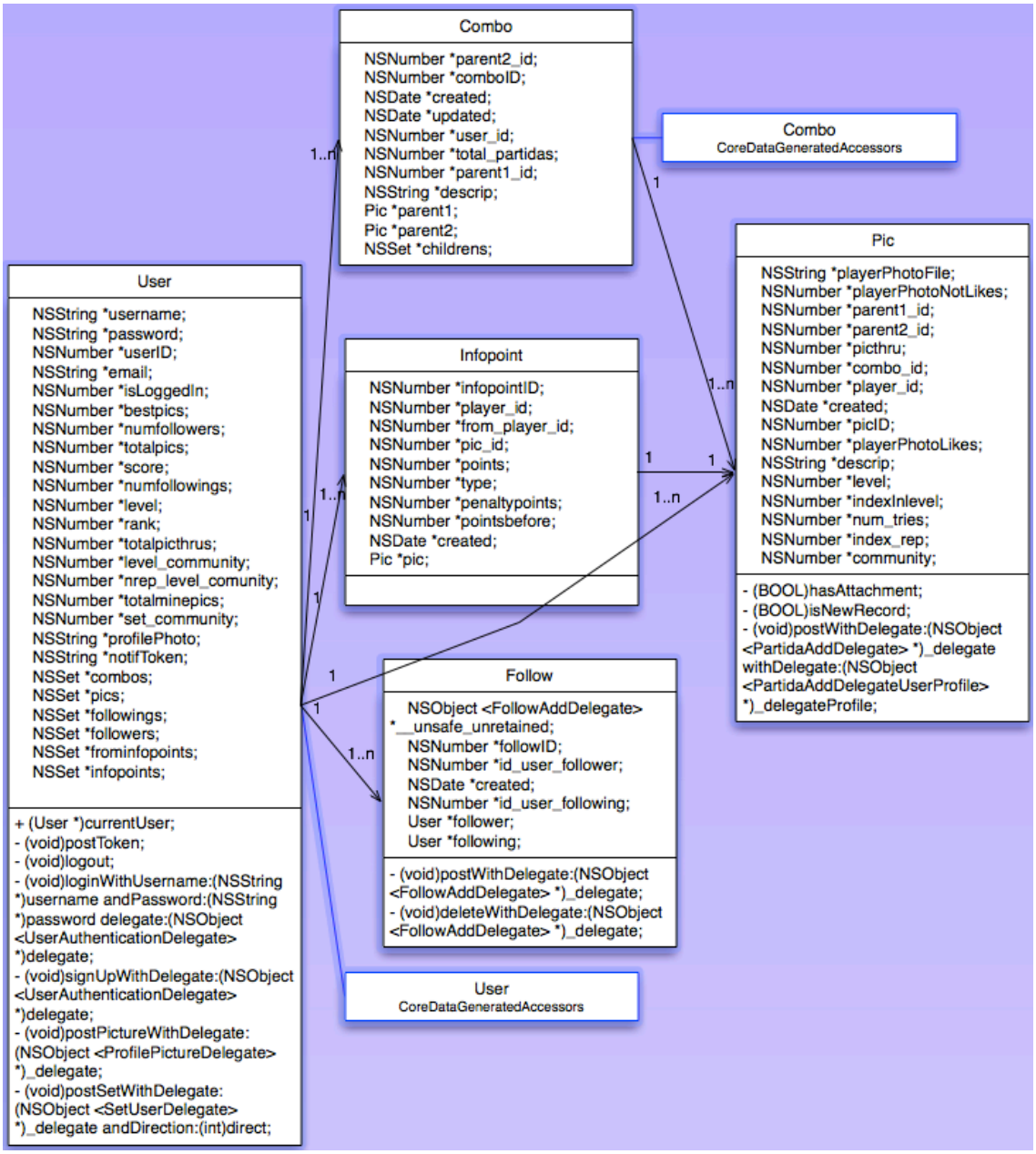


Figura 7

Como vamos a trabajar con los datos que recibimos y enviamos en el formato JSON, hemos utilizado la clase de SBJSON que se puede ver en la figura 8.

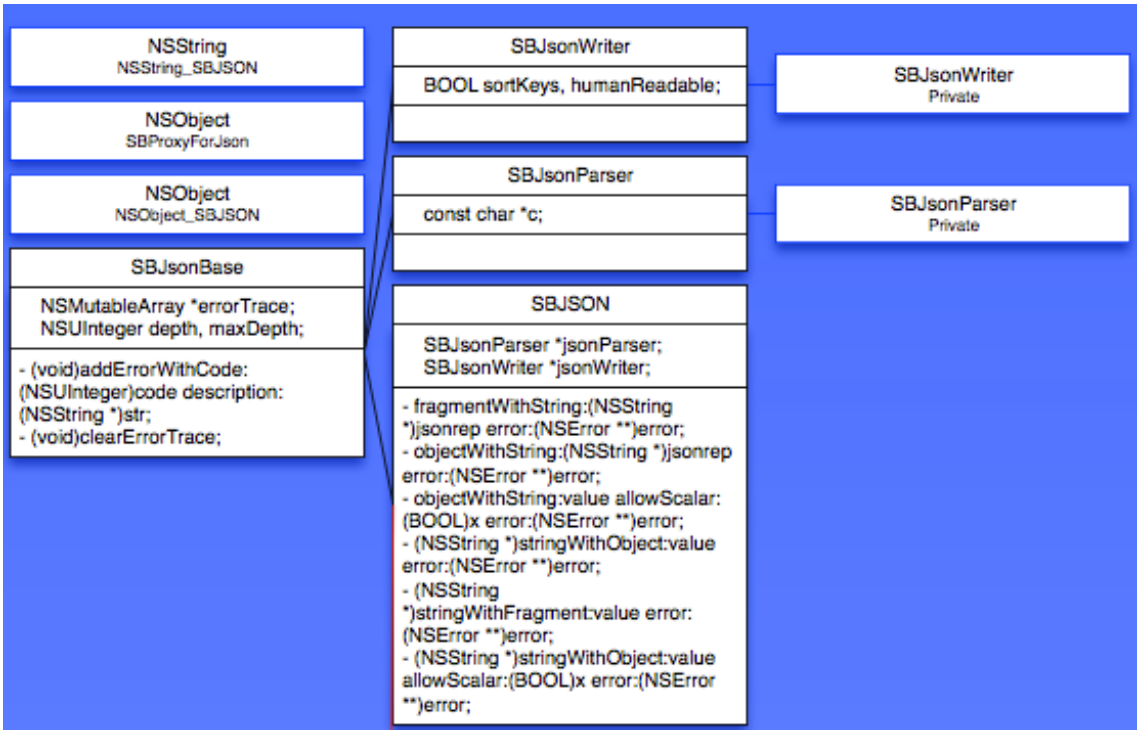


Figura 8

Separación por módulos

Vamos a utilizar la metodología funcional. Esta metodología trata de transformar las especificaciones del problema en una serie de módulos de forma que cada uno realice una función específica y al ejecutarse resuelven el problema.

Así pues vamos a dividir todas las funciones que deben existir en módulos:

Modulo de acceso Este modulo contendrá funciones que están dirigidas a controlar el acceso de usuarios al sistema.

Modulo de administración: Estas funciones son para el administrador y le permiten eliminar cualquier usuario o foto. Además el administrador podrá realizar la selección de las mejores fotos a combinar en el siguiente nivel de cada conjunto.

Modulo de gestión de fotos: Estas funciones son para hacer y subir las fotos al servidor y a Facebook. Además incluye los funciones que se encargaran de guardar las fotos y luego mostrar todo lo que hay a un usuario. También incluye funciones para votar por las mejores fotos y crear con ellas nuevas combinaciones.

Modulo de gestión de usuario: Estas funciones son para la gestión de los usuarios. Además incluye las funciones que se encargaran de preparar toda la información necesaria para visualizar los perfiles de los usuarios. También incluye funciones para poder seguir a los usuarios.

Modulo de notificaciones: Este módulo se encarga de notificar a los usuarios cuando estos ganan puntos. Así pues el usuario recibirá una notificación cuando alguien empieza a seguirlo o cuando a alguien le gusta su foto. También recibirá una notificación cuando su foto se convierta en pichtru, es decir, la mejor foto para una combinación.

El diagrama de módulos es la figura 9.

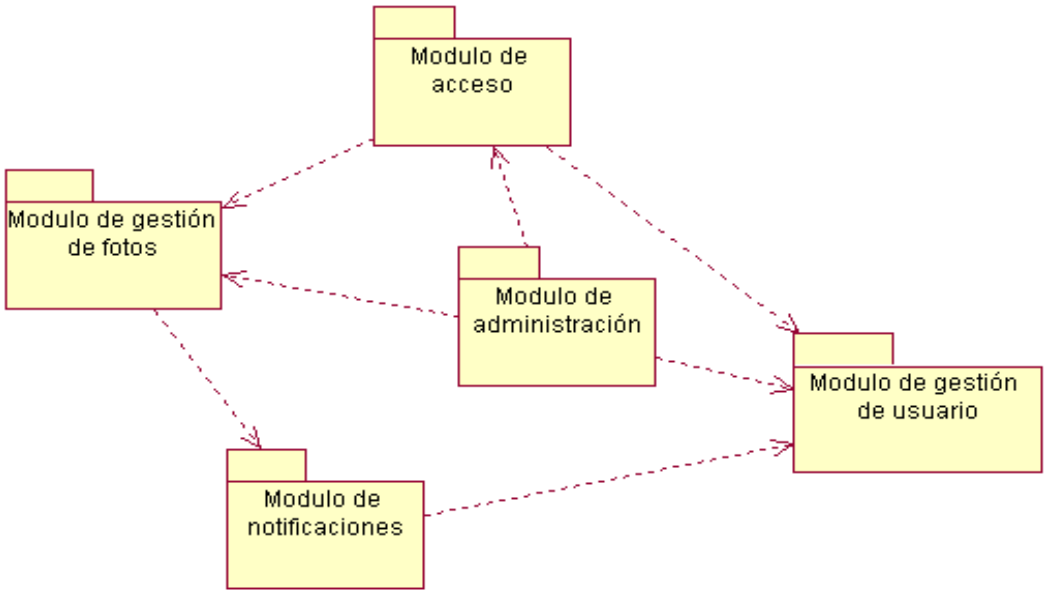


Figura 9

Diagrama Entidad-Relación

Es evidente que tenemos que almacenar información en algún lugar. Como la base de datos es una solución más útil y sencilla que el uso de ficheros, se decidió que se usaría una base de datos. La base de datos debe ser alojada en un servidor ya que la aplicación es de tipo online.

Después de realizar el mapeo de clases a modelo relacional, obtenemos el siguiente diagrama de entidad – relación(ver la figura 10 en la pag. 21) .

Las entidades son:

User: Tendrá sus datos relacionados y podrá crear las fotos(pics) y combinaciones (combos). También tendrá almacenados seguidores y a los usuarios que esta siguiendo(followers y followings).

Follow: Aquí estarán guardados seguidores de los usuarios.

Infopoint: Es la entidad en la cual un usuario tendrá guardada la información relacionada con sus puntuaciones.

Pic: Es la entidad agregada por el usuario que tendrá el id único de la foto y su información asociada.

Combo: Aquí estarán guardadas las combinaciones creadas por los Usuarios.

Las relaciones existentes:

Creado combos: Esta relación es de 1:N porque un usuario podrá tener muchas combinaciones creadas por el. La relación inversa de esta relación es **creado por usuario** en la entidad combo.

Creado pics: Un usuario puede tener muchas fotos subidas por el. La relación inversa de esta relación es **creado por usuario** en la entidad pic.

Tiene followers: Los usuarios podrán tener muchos seguidores. La relación inversa de esta relación es **user_follower** en la entidad follow.

Tiene followings: Un usuario puede seguir a N usuarios. La relación inversa de esta relación es **user_following** en la entidad follow.

Tiene infopoints: Los usuarios tendrán información relacionada con sus puntuaciones. La relación inversa de esta relación es **pertenece a usuario** en la entidad infopoint.

Relacionado con pic: Cada información estará asociada con una foto. La relación inversa de esta relación es **esta en infopoints** en la entidad pic.

Adjudica infopoints: Un usuario adjudicará los puntos a otro usuario realizando alguna actividad como seguir a un usuario o votar por una foto. La relación inversa de esta relación es **adjudicado por usuario** en la entidad infopoint.

Formado por pic1 y formado por pic2: Una combinación estará formados por dos fotos. La relación inversa de esta relación es **usado para formar combos 1 y 2** en la entidad pic.

Pics relacionados: Una combinación tendrá asociada N fotos. La relación inversa de esta relación es **pertenece a combo** en la entidad pic.

Así en el figura 8 están todos las entidades y relaciones que forman el diagrama de entidad – relación de esta aplicación.

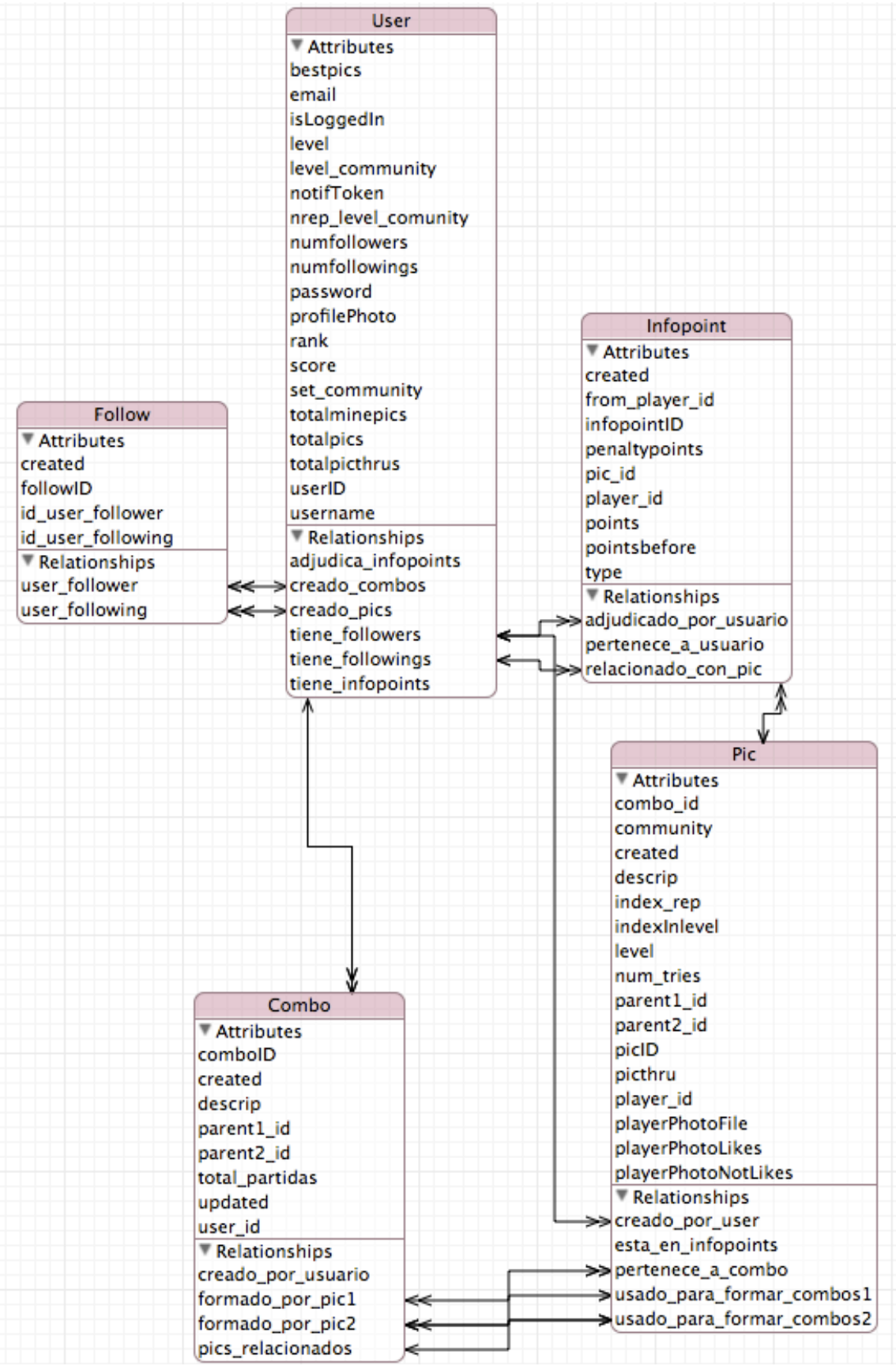


Figura 10

3.1.2 DISEÑO E IMPLEMENTACIÓN

En este apartado se realizara el diseño del sistema y después se decidirán los pasos a seguir para implementar el código fuente.

Paso a tablas del diagrama ER

Vamos a describir las tablas que hemos conseguido a partir del diagrama entidad - relación:

CAKE_SESSIONS

Field	Type	Null	Default
id	varchar(255)	No	
data	text	Yes	*NULL*
expires	int(11)	Yes	*NULL*

Esta tabla guardará todas los sesiones iniciadas del sistema, lo cual soluciona el problema de las sesiones temporales de CakePHP. Las sesiones de CakePHP solo duraban media hora y con esta tabla hemos prolongado el tiempo de vida de cada sesión hasta varios años. También guardamos la fecha de creación de cada sesión y la fecha de su expiración.

USERS

Field	Type	Null	Default
id	int(11)	No	
username	varchar(50)	No	
password	varchar(40)	Yes	*NULL*
score	int(11)	No	0
email	varchar(50)	No	
profilePhoto	varchar(50)	Yes	*NULL*
level	int(10)	No	1
level_community	int(10)	No	1
set_community	int(11)	No	1
last_played_level_set1	int(11)	No	1
nrep_level_comunity	int(10)	No	0
num_pic_level_community	int(10)	No	0
created	datetime	Yes	*NULL*
ip	varchar(100)	Yes	*NULL*
country	varchar(200)	Yes	*NULL*
date_last_login	datetime	Yes	*NULL*
date_last_logout	datetime	Yes	*NULL*
notifToken	varchar(100)	Yes	*NULL*

Esta tabla contiene la información de todos los usuarios registrados del sistema. Guardamos los datos de usuario como el nombre de usuario, contraseña, número de puntos, correo electrónico y la url de su foto de perfil. También guardamos el token del dispositivo de usuario para poder enviarle las notificaciones. Los datos como ip, país y

la fecha de último login realizado sirven para hacer las estadísticas. Otros datos como nivel y conjunto se usan para saber en que conjunto y nivel se encuentra un usuario.

COMBOS

Field	Type	Null	Default
id	int(10)	No	
parent1_id	int(10)	No	0
parent2_id	int(10)	No	0
user_id	int(10)	Yes	*NULL*
created	datetime	No	0000-00-00 00:00:00
updated	datetime	No	0000-00-00 00:00:00
description	varchar(140)	No	

Esta tabla contiene todas las combinaciones creadas por los usuarios con el identificador user_id. Todas las combinaciones están formadas por dos fotos con identificadores parent1_id y parent2_id. También guardamos la fecha de creación y de última actualización de cada combinación.

FOLLOWS

Field	Type	Null	Default
id	int(10)	No	
id_user_follower	int(10)	No	0
id_user_following	int(10)	No	0
created	datetime	No	0000-00-00 00:00:00

Esta tabla guarda información básica sobre los seguidores de los usuarios. Así pues los registros están compuestos por un identificador de usuario id_user_follower, que está siguiendo al usuario con el identificador id_user_following.

PICS

Field	Type	Null	Default
id	int(10)	No	
player_id	int(10)	Yes	*NULL*
combo_id	int(10)	No	0
parent1_id	int(10)	Yes	*NULL*
parent2_id	int(10)	Yes	*NULL*
playerPhotoFile	varchar(50)	Yes	*NULL*
playerPhotoLikes	int(10)	No	0
playerPhotoNotLikes	int(10)	No	0
created	datetime	Yes	*NULL*
description	varchar(140)	Yes	*NULL*
picthru	int(1)	No	0
timeschoosen	int(10)	No	0
level	int(10)	Yes	*NULL*
indexInlevel	int(10)	Yes	*NULL*
choosingIndexInLevel	int(10)	No	0
community	tinyint(1)	No	1
rep_num	int(6)	No	0
index_rep	int(6)	No	0
num_tries	int(11)	No	0

Esta tabla contiene todas las fotos creadas por los usuarios con el identificador player_id. Todas las fotos están asociadas a una combinación con el identificador comdo_id. También guardamos otros datos de las fotos como el número de votos, la url y la fecha de creación.

INFOPOINTS

Field	Type	Null	Default
id	int(11)	No	
player_id	int(11)	No	0
created	datetime	No	0000-00-00 00:00:00
type	int(3)	No	0
points	int(11)	Yes	0
pointsbefore	int(11)	No	0
penaltypoints	int(11)	No	0
pic_id	int(11)	Yes	*NULL*
from_player_id	int(11)	Yes	*NULL*

Esta tabla guarda información sobre las puntuaciones de usuario con el identificador player_id. Así pues los registros están compuestos por el identificador pic_id de la foto que ha recibido el voto y el identificador de usuario que ha adjudicado los puntos.

INFOSETS

Field	Type	Null	Default
id	int(11)	No	1
level	int(11)	No	1
date_modified	date	No	0000-00-00

Esta tabla contiene la información de cada conjunto de las fotos con las que se hacen los combinaciones. El atributo nivel guarda el máximo nivel de cada conjunto.

Como resultado tendremos en la carpeta models de cakephp un archivo para cada tabla. En estos archivos se describe el modelo de datos. Por ejemplo para las tablas users y pics tendremos los siguientes archivos:

```
class User extends AppModel {
   var $name = 'User';

   var $hasMany = array(
                'Pic' => array(
                        'className' => 'Pic',
                        'foreignKey' => 'player_id'
                        ),
                'Combo' => array(
                        'className' => 'Combo',
                        'foreignKey' => 'user_id'
                        )
        );
}

class Pic extends AppModel {
   var $name = 'Pic';

   var $belongsTo = array(
                'Combo' => array(
                        'className' => 'Combo',
                        'foreignKey' => 'combo_id'
                        ),
                 'User' => array(
                        'className' => 'User',
                        'foreignKey' => 'player_id',
                        'fields' => array('id', 'username', 'score', 'email', 'profilePhoto',
                        'level', 'level_community', 'nrep_level_comunity',
                        'num_pic_level_community')
    )
);
```

Una relación hasMany indica que un usuario puede tener muchas pics(fotos) y combos(combinaciones) creados por él. Una relación belongsTo indica que un pic pertenece a un solo usuario. Cakephp necesita esta descripción para crear las respuestas JSON a las consultas solicitadas desde el iPhone.

Elección de la tecnología a usar

En ese apartado se indicará que herramientas, librerías y lenguajes de programación fueron usados para implementar la aplicación.

Esta aplicación se realizo en un ordenador (Servidor) equipado con:

Sistema Operativo X Lion
Xcode
APNS
phpMyAdmin
Apache
CakePHP
Fugu

Xcode

Xcode es el entorno de desarrollo integrado (IDE, en sus siglas en inglés) de Apple Inc. y se suministra gratuitamente junto con Mac OS X. Xcode trabaja conjuntamente con Interface Builder, una herencia de NeXT, una herramienta gráfica para la creación de interfaces de usuario.

Xcode incluye la colección de compiladores del proyecto GNU (GCC), y puede compilar código C, C++, Objective-C, Objective-C++, Java y AppleScript mediante una amplia gama de modelos de programación, incluyendo, pero no limitado a Cocoa, Carbón y Java. Otras compañías han añadido soporte para GNU Pascal,1 Free Pascal,2 Ada y Perl.3

APNS Apple Push Notification Service

El **Apple Push Notification Service** (servicio de alertas Apple Push Notification) es un servicio móvil creado por la compañia Apple Inc. El servicio usa la tecnología Push a través de una conexión IP constantemente abierta para enviar notificaciones de los servidores de aplicaciones de terceros para el iPhone, iPod Touch y ahora también el iPad. Las notificaciones pueden incluir logos, canciones o alarmas de texto personalizadas. En figura 11 se puede ver el esquema de mecanismo de funcionamiento de apns.

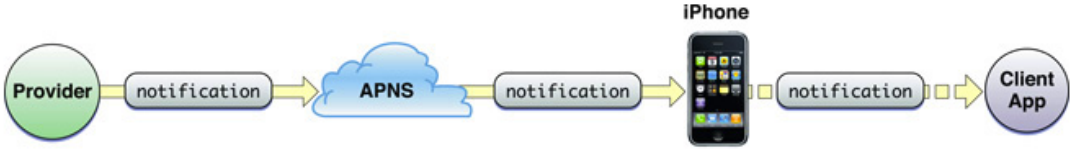


Figura 11

En la figura 12 se puede ver un ejemplos de recepción de notificación.

Figura 12

PhpMyAdmin

Para trabajar con la base de datos hemos elegido MySQL. Porque es muy fácil de usar, además hay una aplicación llamada phpmyAdmin que facilita el trabajo con MySQL y es muy útil para gestionar la base de datos. MySQL es un sistema de gestión de bases de datos relacionales, multihilo y multiusuario con más de seis millones de instalaciones. MySQL es muy utilizado en las aplicaciones web, su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

Apache

También hemos elegido como servidor Apache, porque Apache presenta entre otras características altamente configurables, las bases de datos de autenticación y negociado de contenido. Apache, es el servidor HTTP más usado.

CakePHP

CakePHP es un marco de desarrollo [framework] rápido para PHP, libre, de código abierto. Se trata de una estructura que sirve de base a los programadores para que éstos puedan crear aplicaciones Web. Con este framework se puede trabajar de forma estructurada y rápida, sin pérdida de flexibilidad.

CakePHP tiene un equipo de desarrolladores y una comunidad activos, lo que añade valor al proyecto. Con CakePHP, además de no tener que reinventar la rueda, el núcleo de cualquier aplicación se mejora constantemente y está bien probado.

Fugu

Fugu es un interface para Mac OS X que permite ejecutar conexiones a servidores FTP, de forma segura, protegiéndolos de los famosos “sniffers”. Fugu trabaja como la mayoría de los clientes gráficos de FTP, excepto a la hora de introducir el “Nombre de usuario” y la “Contraseña”, y los archivos, ya que los encripta de forma automática.

Hemos utilizado las siguiente librerías:

RestKit

RestKit es una biblioteca de gran alcance que simplifica la interacción con los servicios web para aplicaciones de IOS. Esta biblioteca fuente madura y abierta que proporciona una capa entre los objetos del dominio de su app y la nube.

RestKit le permite trabajar con los objetos de cocoa, y después serializarlos a y desde su sistema backend.

RestKit también proporciona un sistema para modelar recursos alejados trazándolos de las cargas útiles de JSON.

FBConnect

API de facebook que nos permite que los usuarios "se identifiquen" en nuestra aplicación con su usuario y contraseña de Facebook. Proporciona muchas funciones para trabajar con Facebook como por ejemplo subir los fotos y comentarios al muro de Facebook.

Rs-SDwebImage

Es una librería para cargar imágenes remótas en iOS. Las imágenes se guardan en la cache para no hacer descargas repetidas.

SFHFKeychainUtils

Es una librería para el acceso a keychain de cocoa/objective-c. Nos facilita el acceso seguro a las contraseñas de usuario guardadas en el keychain.

Módulos

En el análisis hemos determinado los módulos y ya sabemos que hace cada uno de ellos. Ahora queda por saber cómo funcionan. Para poder verlo hay que describir cada una de las funciones de cada modulo. Los módulos y las funciones menos significativas pueden ser omitidas.

MÓDULO DE ACCESO

El acceso a la aplicación se realiza a través de las pantallas de la imágenes 1 y 2.
El funcionamiento del registro es el siguiente. Se crea un objeto de clase User y luego se serializa con el servidor por Restkit. Los datos de usuario se pasan al servidor con la petición de POST en el url http://app.picthru.net/cakepicthru2/users/registrar.json. 'Registrar' es una función escrita en php que se encuentra en el servidor. Cuando recibe los datos de nuevo usuario, crea un nuevo registro para este usuario en la base de datos y envía como respuesta los datos completos de este usuario en formato JSON. La respuesta de esta función es la que sigue:

```
   User =     {
      country = "<null>";
      created = "2012-07-17 11:07:15";
      "date_last_login" = "<null>";
      "date_last_logout" = "<null>";
      email = "jon@unavarra.es";
      id = 176;
      ip = "<null>";
      isLoggedIn = 1;
      "last_played_level_set1" = 3;
      level = 1;
      "level_community" = 3;
      notifToken = "<null>";
      "nrep_level_comunity" = 0;
      "num_pic_level_community" = 0;
      password = "<null>";
      profilePhoto = "<null>";
      score = 0;
      "set_community" = 1;
      username = jon;
   };
}
```

En el lado del cliente, iPhone de usuario, se recibe la respuesta de esta función y se mapea en el objeto de usuario actual de la aplicación por el Restkit.
El mecanismo de funcionamiento de la aplicación con el servidor es muy similar a la explicación anterior.

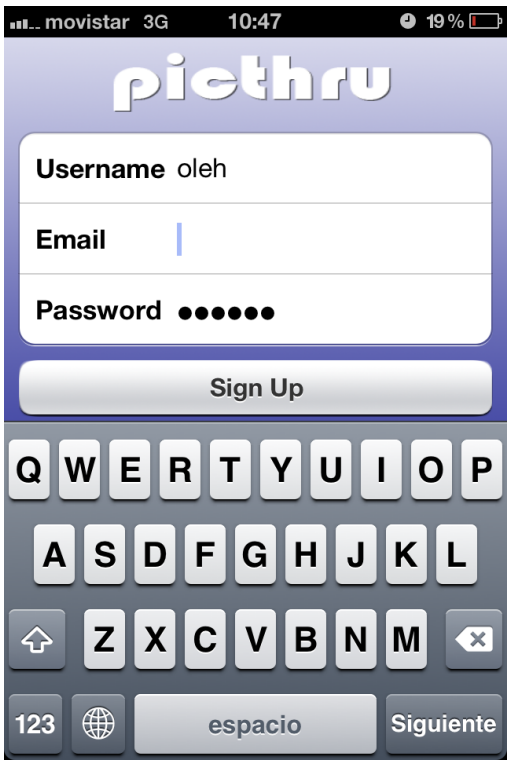


Imagen 1

Imagen 2

A partir de aquí se explicaran las funciones que tiene este modulo:

Llamada desde	Desde Restkit
Función	objectLoader
Entrada	(RKObjectLoader*)objectLoader didLoadObjects:(NSArray *)objects
Salida	void
Descripción	Esta función se invoca desde el Restkit en caso en que los objetos recibidos de servidor en formato JSON se han mapeado correctamente. Con la función wasSentToResourcePath podemos saber a que url se ha hecho la petición.
Errores	En caso de error de mapeo de los objetos el Restkit llamara a -(void)objectLoader:(RKObjectLoader *)objectLoader didFailWithError:(NSError*)error que debe ser implementado por el delegado.

Llamada desde	LoginSignUpViewController es Clase controlador de pantalla de imagen 1.
Función	signUpWithDelegate
Entrada	(NSObject<UserAuthenticationDelegate> *)delegate
Salida	void
Descripción	Esta función es para registrarse en la aplicación. El registro se realiza a través de llamada POST al servidor Apache en url users/registrar.json. Si el registro se ha realizado correctamente entonces el servidor responde con el objeto de nuevo usuario registrado en el formato json, si no la respuesta es el error en formato json. Cuando el mapeo de los datos JSON en elos objetos se ha realizado correctamente el Restkit llama a la -(void)objectLoader:(RKObjectLoader*)objectLoader didLoadObjects:(NSArray *)objects Si el registro se realizado correctamente esto significa que ya estamos también logeados en el sistema, así pues se llama a la función loginWasSuccessful.

Errores	En caso de error el servidor responde en formato de JSON asi:
	errors:{
	This user name exists
	}

Llamada desde	LoginSignUpViewController es Clase controlador de pantalla de imagen 1.
Función	loginWithUsername
Entrada	(NSString*)_username andPassword:(NSString*)_password delegate:(NSObject<UserAuthenticationDelegate>*)_delegate
Salida	void
Descripción	Esta función es para hacer log in en la aplicación. El registro se realiza a través de llamada POST al servidor Apache en url users/loginapp.json. Si el login se ha realizado correctamente entonces el servidor responde con el objeto de nuevo usuario logeado en el formato de json sino

	la respuesta es el error en formato json. Cuando el mapeo de los datos JSON en los objetos se ha realizado correctamente el Restkit llama a la -(void)objectLoader:(RKObjectLoader*)objectLoader didLoadObjects:(NSArray *)objects Si el login se ha realizado correctamente, en el servidor se había iniciado la sesión de usuario, y seguidamente se realiza la llamada a la función loginWasSuccessful.

Errores	En caso de error el servidor responde en formato de JSON asi:
	errors:{
	Invalid username or password
	}

Llamada desde	LoginSignUpViewController es Clase controlador de pantalla de imagen 1.
Función	logout
Entrada	
Salida	void
Descripción	Esta función es para registrarse en la aplicación. El registro se realiza a través de llamada al servidor Apache en url users/ logoutapp.json. Se cierra la sesión de usuario en el servidor.

Llamada desde	LoginSignUpViewController es Clase controlador de pantalla de imagen 1.
Función	loginWasSuccessful
Entrada	
Salida	void
Descripción	Se inicializa el usuario actual de sistema con la función setCurrentUser de la clase User.

MODULO DE GESTION DE FOTOS

En la imagen 3 se puede ver la pantalla principal para crear nuevas combinaciones. Antes de enseñar esta pantalla se llaman a las funciones que traen los datos de servidor. A partir de aquí se explicaran estas funciones:

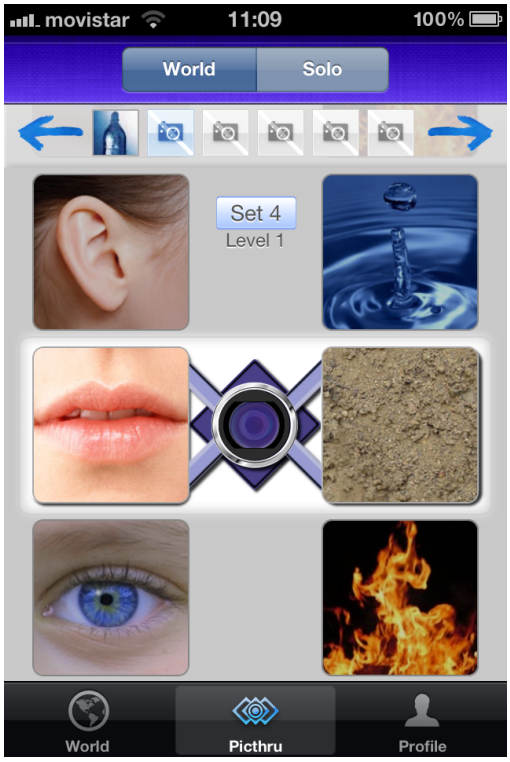


Imagen 3

Llamada desde	Método viewDidLoad de controlador LevelCombine de pantalla de imagen 3.
Función	**comunitypicthrus**
Entrada	
Salida	Imágenes en formato JSON
Descripción	Esta función se encuentra en el servidor y es para consultar las imágenes que hay que enseñar en la pantalla. La respuesta de la función contiene la información de estas imágenes como el url y su id.

Llamada desde	El método implementado de delegado de Restkit objectLoader de controlador LevelCombine de pantalla de imagen 3.
Función	loadImages
Entrada	(NSArray*)arrImages
Salida	void
Descripción	Esta función es para descargar y enseñar las imágenes de arrImages de forma asíncrona.

Llamada desde	El controlador LevelCombine de pantalla de imagen 3.
Función	spin
Entrada	(id)sender
Salida	IBAction
Descripción	Esta función sirve para elegir la combinación de las imágenes. Mueve las imágenes aleatoriamente de izquierda y derecha cuando hacemos un click sobre ellas.

Llamada desde	El controlador LevelCombine de pantalla de imagen 3.
Función	switchToPlay
Entrada	
Salida	IBAction
Descripción	Determina las imágenes seleccionadas y abre la pantalla de la imagen 4.

En la pantalla de la imagen 4 se puede observar como se hace una foto para una combinación y en la la pantalla de la imagen 5 se puede observar como se sube una foto al servidor o Facebook.

Imagen 4

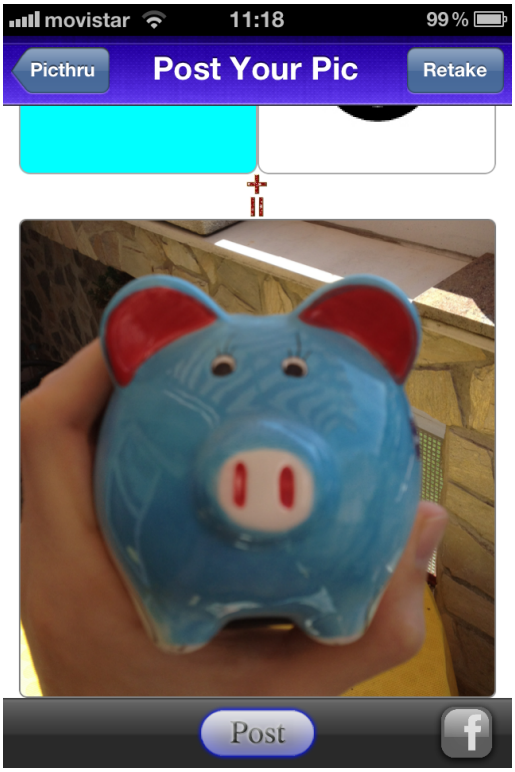


Imagen 5

Estas funciones son para hacer y subir las fotos a servidor y a Facebook:

Llamada desde	El controlador CameraViewController de pantalla de imagen 4.
Función	imagePickerController
Entrada	(UIImagePickerController *)pickerL didFinishPickingImage:(UIImage *)image editingInfo:(NSDictionary *)info
Salida	void
Descripción	Guarda la imagen que corresponde al área dentro del cuadrado de la cámara. La imagen se guarda en el álbum y se prepara para el envío al servidor.

Llamada desde	El controlador CameraViewController de pantalla de imagen 4.
Función	scaleAndRotateImage
Entrada	(UIImage*) image
Salida	UIImage
Descripción	Escala y rota una foto. Nos hace falta esta función para redimensionar las fotos a tamaño 600x600. También hace falta rotar la foto en caso si fue tomada con el angulo menor a -90 o mayor a 90.

Llamada desde	El controlador CameraViewController de pantalla de imagen 4.
Función	post
Entrada	(id)sender
Salida	IBAction
Descripción	Esta función sirve para subir la foto al servidor. La subida se realiza mediante la petición POST.

Llamada desde	El controlador CameraViewController de pantalla de imagen 4.
Función	apiDialogFeedUser
Entrada	
Salida	IBAction
Descripción	Esta función sirve para subir la foto al muro de Facebook.

El modulo también incluye las funciones que se encargaran de mostrar todas las fotos a los usuarios. En la pantalla de la imagen 6 se pueden ver las fotos hechas por los

usuarios.

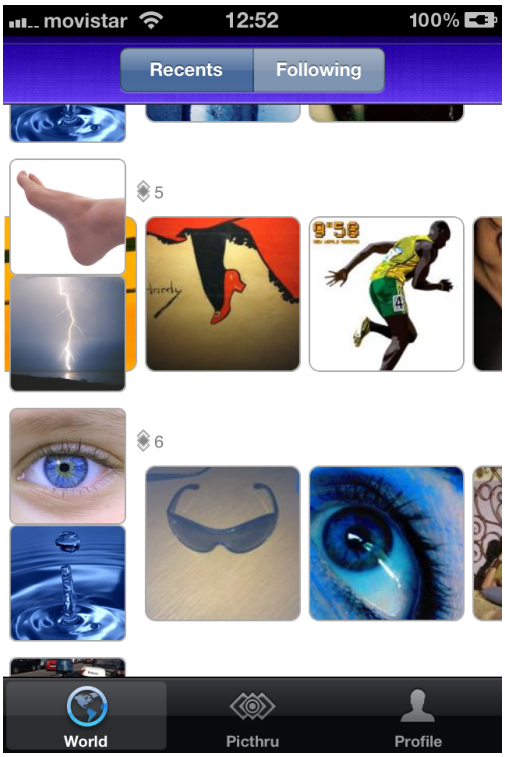


Imagen 6

Estas funciones descargan las fotos y las enseñan a los usuarios:

Llamada desde	Método viewDidLoad de controlador LevelsViewController de pantalla de imagen 6.
Función	latestcombos
Entrada	Número de página
Salida	Combinaciones con las imágenes en formato JSON
Descripción	Esta función se encuentra en el servidor y sirve para consultar las combinaciones que hay que enseñar en una pagina de la pantalla. La respuesta de la función contiene la información de estas combinaciones y las fotos.

Llamada desde	El método implementado de delegado de Restkit objectLoader de controlador controlador LevelsViewController de pantalla de imagen 6.
Función	makeView
Entrada	(NSArray*) combos
Salida	void
Descripción	Esta función sirve para descargar y

	enseñar las combinaciones de forma asíncrona.

Llamada desde	El controlador LevelsViewController de pantalla de imagen 6.
Función	showViewPartida
Entrada	(int)partidaID
Salida	void
Descripción	Abre la pantalla que enseña la foto en tamaño grande y toda la información relacionada con esta foto.

Llamada desde	El controlador LevelsViewController de pantalla de imagen 6.
Función	updatePositionsImages
Entrada	(int)dist
Salida	void
Descripción	Actualiza la posición de las fotos movidas por los usuarios.

MODULO DE NOTIFICACIONES

Este módulo se encarga de notificar a los usuarios. El modulo fue implementado usando el proveedor de notificaciones Urban Airship. Para poder enviar las notificaciones hace falta tener el token de dispositivo. En la figura 13 se puede ver como se solicita el token de un dispositivo en el servicio APNS de Apple y se almacena en el proveedor.

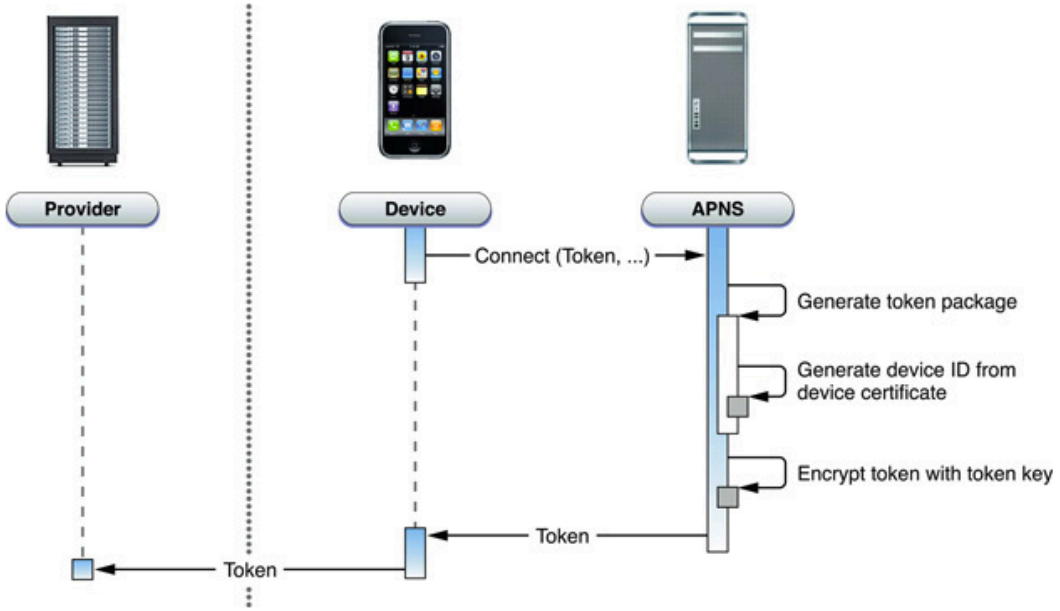


Figura 13

Posteriormente el dispositivo se conecta con su token a los servidores APNS de Apple. El proveedor envía las notificaciones a través de los servidores APNS a los dispositivos usando el token como identificador de dispositivos. En la figura 14 se puede ver este proceso.

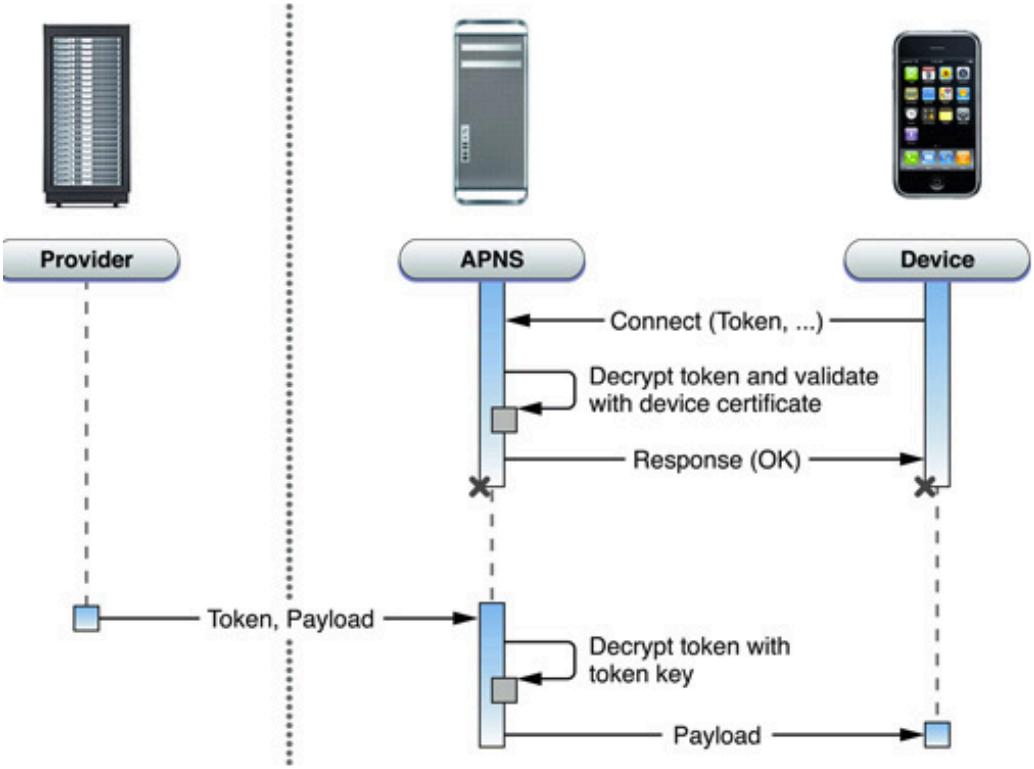


Figura 14

4. ANÁLISIS DE RESULTADOS

Cuando la implementación fue finalizada, se realizó la prueba completa del sistema con los datos reales. Este paso permite una comparación precisa de la salida del nuevo sistema con a que sabe que es salida correctamente procesada, así como una buena aproximación de cómo serán manejados los datos reales.

Para probar el sistema se hicieron diferentes tipos de pruebas:

Pruebas de unidad: a medida que se hacían los módulos se probaba que funcionasen de forma individual correctamente. Durante esta fase se corrigieron muchos errores de todo tipo.

Pruebas de diseño: una vez comprobado el correcto funcionamiento individualmente, se paso a conectar los módulos. Se corrigieron algunos errores.

Pruebas de validación y sistema: se corrigieron errores y se probó que el proyecto cumplía con todos los requisitos.

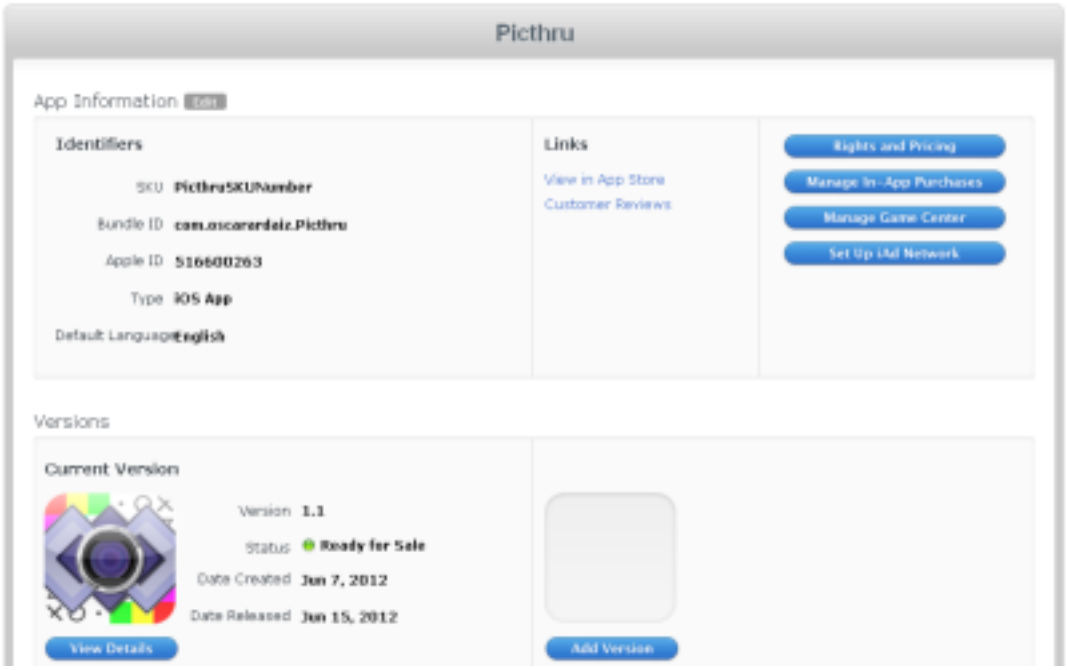


Tabla 1

Finalmente la aplicación fue revisada y aprobada por el equipo de AppStore. En la tabla 1 se puede observar la fecha de finalización del desarrollo de la aplicación, asi como la de aprobación de la misma por parte del equipo de la AppStore.

Igualmente, en la tabla 2 se puede observar el numero de descargas en la primera semana en AppStore ordenada por países.

Dashboard | Sales

View : Daily Weekly | Week : Apr 23 to Apr 29, 2012 | Download Report: Sales

Free Apps | Paid Apps | In Apps | Updates

Title	Developer	Version	Type	Units	Customer Price	Proceeds	Store	Apple ID
Picthru	Oscar Ardaiz	1.0	1	21	0 USD	0 USD	TH	516600263
Picthru	Oscar Ardaiz	1.0	1	15	0 USD	0 USD	US	516600263
Picthru	Oscar Ardaiz	1.0	1	9	0 EUR	0 EUR	ES	516600263
Picthru	Oscar Ardaiz	1.0	1	6	0 CAD	0 CAD	CA	516600263
Picthru	Oscar Ardaiz	1.0	1	5	0 JPY	0 JPY	JP	516600263
Picthru	Oscar Ardaiz	1.0	1	5	0 AUD	0 AUD	AU	516600263
Picthru	Oscar Ardaiz	1.0	1	5	0 MXN	0 MXN	MX	516600263
Picthru	Oscar Ardaiz	1.0	1	5	0 CNY	0 CNY	CN	516600263
Picthru	Oscar Ardaiz	1.0	1	4	0 USD	0 USD	SG	516600263
Picthru	Oscar Ardaiz	1.0	1	4	0 GBP	0 GBP	GB	516600263
Picthru	Oscar Ardaiz	1.0	1	4	0 USD	0 USD	TW	516600263
Picthru	Oscar Ardaiz	1.0	1	3	0 NOK	0 NOK	NO	516600263
Picthru	Oscar Ardaiz	1.0	1	3	0 EUR	0 EUR	FR	516600263
Picthru	Oscar Ardaiz	1.0	1	3	0 USD	0 USD	PH	516600263
Picthru	Oscar Ardaiz	1.0	1	3	0 USD	0 USD	KW	516600263
Picthru	Oscar Ardaiz	1.0	1	3	0 USD	0 USD	ID	516600263

Tabla 2

La evaluación de descargas hechas por los usuarios hasta ahora aparecen en la figura 15.

Figura 7

Como conclusión, el numero de descargas se han bajado. Como cualquier producto en el mercado, para ser vendido bien, necesita ser promocionado anteriormente al publico. Para subir el numero de descargas tenemos que invertir algo en la promoción de la aplicación.

Como resultado en las ilustraciones 2, 3, 4 y 5 se puede ver algunas fotos creativas subidas por los usuarios en los diferentes niveles.

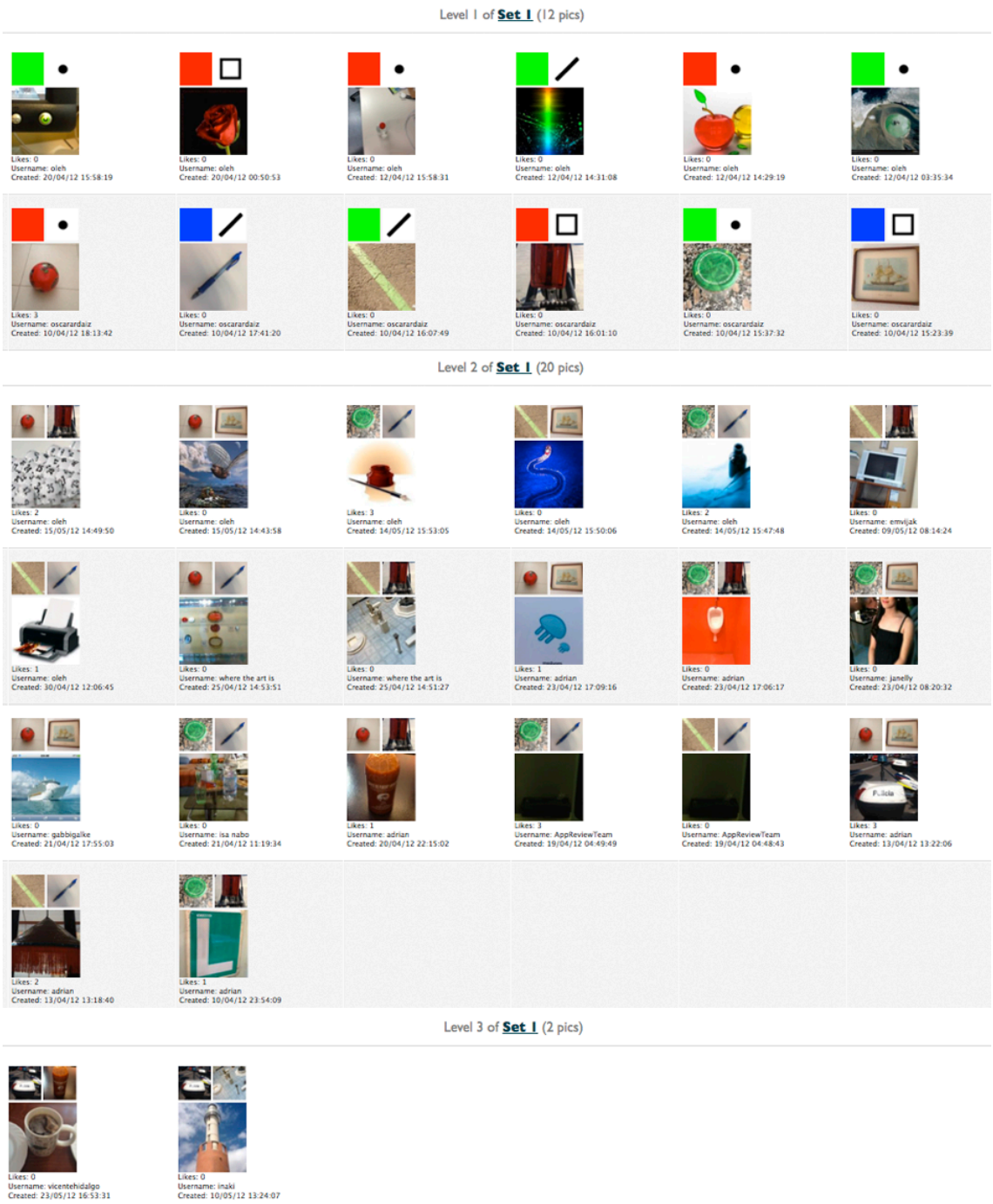


Ilustración 2

En la primera parte de la ilustración 2 aparecen las fotos del conjunto 1. Como se puede observar en este conjunto hay 3 niveles. Con las mejores fotos de nivel 1 se han hecho fotos en el nivel 2. En el nivel 3 también se han usado las mejores fotos de nivel 2.

Las fotos de ilustración 3 son del conjunto 3 y las fotos de ilustración 4 del conjunto 4.

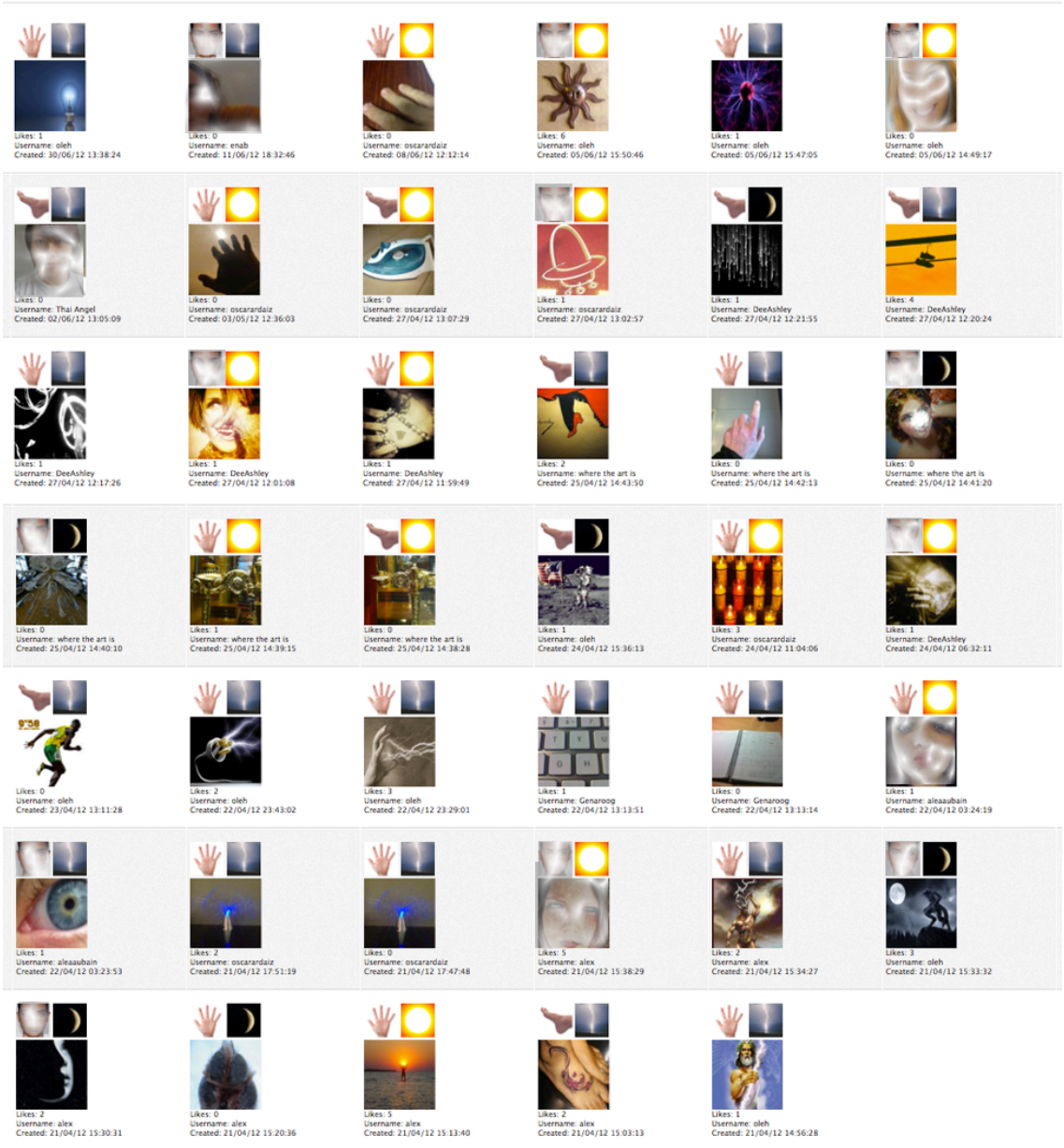


Ilustración 3

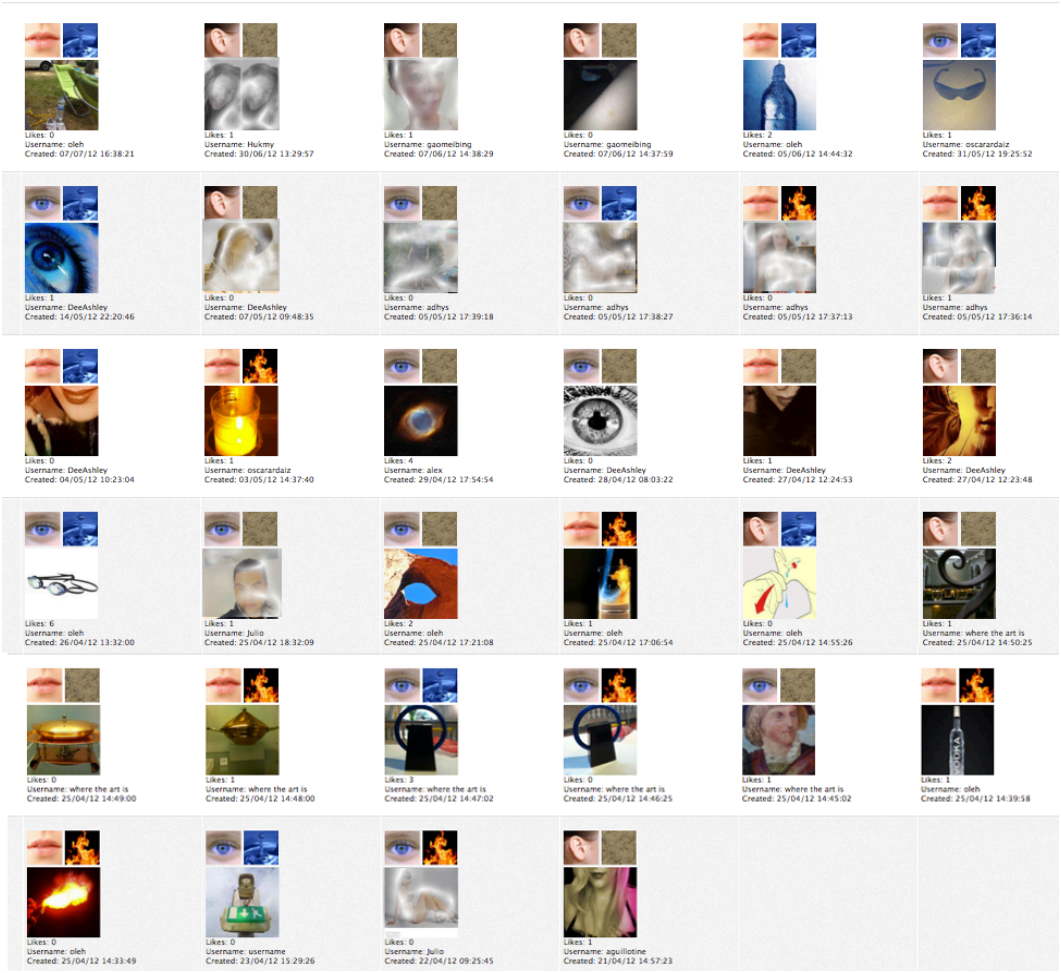


Ilustración 4

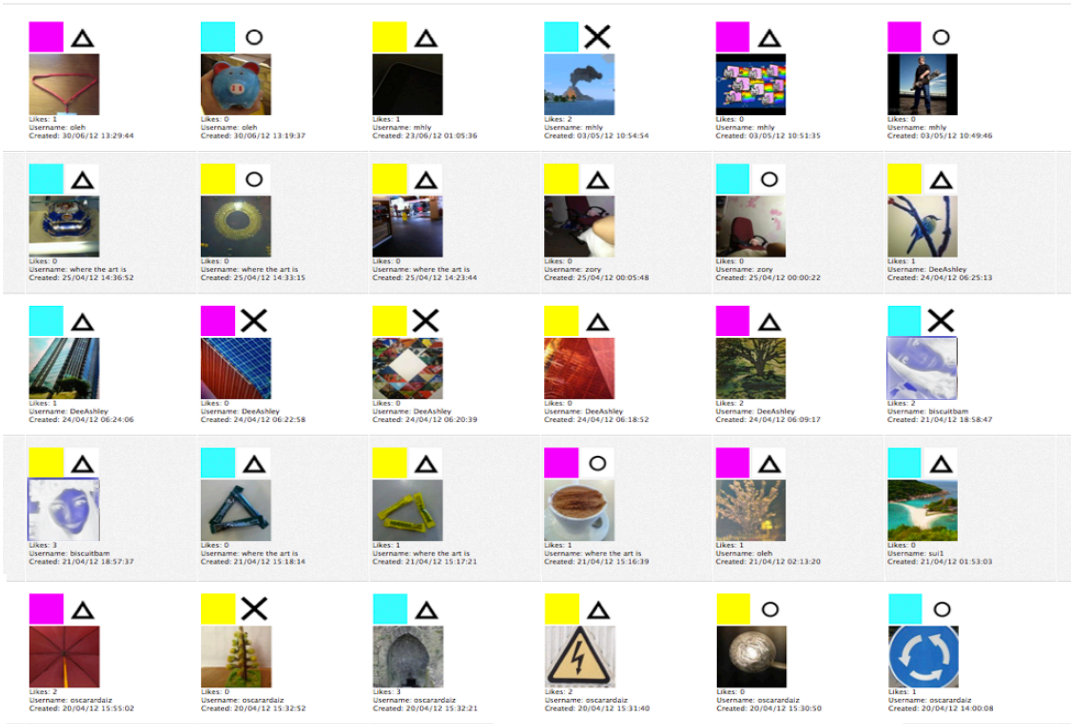


Ilustración 5

5. COSTE Y PLANIFICACIÓN

5.1 PLANIFICACIÓN

A continuación se detallara la organización y el tiempo del trabajo realizado para el análisis, diseño y la implementación. La organización del trabajo se divide en dos partes:
Trabajo de analista: determinar los requisitos, estudiar el entorno de trabajo, elegir el software a usar y diseñar el sistema.
Trabajo de programador: codificación y pruebas.
La planificación de tiempo de proyecto se muestra en la figura 14 y la representación mediante diagrama de Gantt en la figura 15.

	Name	Duration	Start	Finish	Predecessors
1	**⊟Análisis**	8 days	**1/9/12 8:00 AM**	**1/18/12 5:00 PM**	
2	Análisis de los aplica	5 days	1/9/12 8:00 AM	1/13/12 5:00 PM	
3	Especificación de req	3 days	1/16/12 8:00 AM	1/18/12 5:00 PM	2
4	**⊟Diseño**	26 days	**1/19/12 8:00 AM**	**2/23/12 5:00 PM**	
5	Diseño de interfaces	15 days	1/19/12 8:00 AM	2/8/12 5:00 PM	3
6	Diseño de interfaces	4 days	2/9/12 8:00 AM	2/14/12 5:00 PM	5
7	Diseño del modelo E	7 days	2/15/12 8:00 AM	2/23/12 5:00 PM	6
8	**⊟Implementación**	90 days	**2/24/12 8:00 AM**	**6/28/12 5:00 PM**	
9	Codificación	90 days	2/24/12 8:00 AM	6/28/12 5:00 PM	7
10	**⊟Pruebas**	6 days	**6/29/12 8:00 AM**	**7/6/12 5:00 PM**	
11	Realización de prueb	6 days	6/29/12 8:00 AM	7/6/12 5:00 PM	9
12	**⊟Documentación**	11 days	**7/9/12 8:00 AM**	**7/23/12 5:00 PM**	
13	Documentación	11 days	7/9/12 8:00 AM	7/23/12 5:00 PM	11

Figura 8

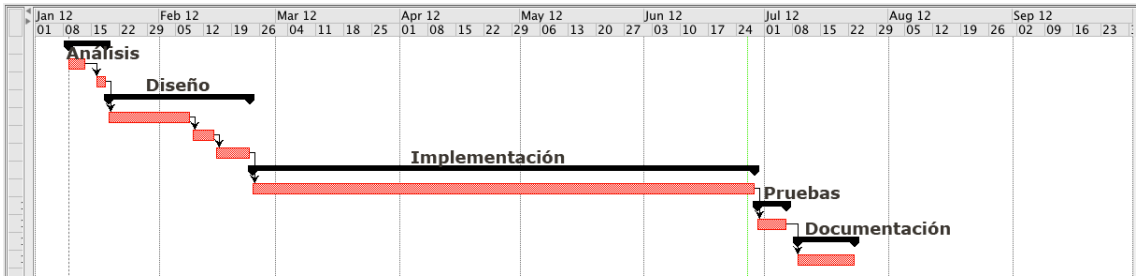


Figura 9

El tiempo total de trabajo realizado es de 144 días (trabajando 6 horas/día), divididas en 90 días de trabajo de programador y 54 días de trabajo de analista.

5.2 COSTES

Se estima que un analista junior cobra 20 euros/hora y un programador junior cobra 15 euros/hora. El trabajo se realizaba aproximadamente en 6 horas por día. El resultado se puede observar en la tabla 3.

Trabajo de	Horas	Euros/hora	Total a pagar en €
Analista	**324**	**17**	**5508**
Programador	**540**	**13**	**7020**
Total	**864**		**12528**

Tabla 3

6. CONCLUSIONES Y PASOS FUTUROS

6.1 CONCLUSIONES

El presente proyecto, que consistía en hacer una aplicación para IOS, la cual permitiese a los usuarios realizar fotos creativas y compartirlas con el resto del mundo empleando un Smartphone, fue realizado satisfactoriamente.

En esta aplicación se pueden crear fotos originales combinado fotos de otros usuarios y ganar puntos si tus fotos son elegidas como Picthru. Las mejores fotos Picthru se convierten en las fotos con que se formaran nuevas combinaciones. Se puede ver el árbol de combinaciones a partir del cual se creó cualquier imagen, eguir a los mejores jugadores para ver sus fotos y competir con tus amigos para demostrar tu creatividad.

Las fotos se toman utilizando la cámara de que dispone el IPhone. Además la aplicación permite a los usuarios elegir y recortar los fotos de los álbumes de sus IPhone. Una vez elegidas y recortadas se suben automáticamente al servidor y todos los demás usuarios pueden ver estas imágenes.

Los usuarios pueden crear fotos originales combinando fotos de otros usuarios y ganar puntos si sus fotos son elegidas como mejores. La foto con mayor número de puntos para una combinación la llamamos Picthru. Las mejores fotos Picthru se coinvertirán en las fotos con que se formarán nuevas combinaciones.

El administrador es capaz de consultar y elegir las mejores fotos a través de la pagina web. Inmediatamente después de la elección de las mejores fotos, los usuarios pueden formar nuevas combinaciones con estas fotos en sus IPhone. En la página web de la aplicación, también puede verse el árbol completo de combinaciones a partir de las cuales se ha creado cualquier imagen. En el IPhone , sin embargo, solo se puede visualizar una parte de este árbol debido a las limitadas dimensiones del dispositivo.

La aplicación dispone de una pantalla con el ranking de los usuarios. En esta pantalla se puede ver una lista de jugadores, ordenada en función del número de puntos que tiene cada uno. De esta manera los jugadores pueden competir con sus amigos para demostrar su creatividad. Además los usuarios pueden seguir a los mejores jugadores para ver sus fotos y adjudicar sus votos a las fotos que prefieran. Las fotos con mayor número de votos se convierten en Picthrus.

Se ha añadido la funcionalidad adicional de recibir notificaciones de actividad dentro de la aplicación. El usuario recibe las notificaciones en el caso de que a alguien le guste su foto o de que un jugador empiece a seguirle.

Además se han añadido las funcionalidades que permiten a los usuarios de la aplicación subir las fotos desde el IPhone directamente a Facebook. Esto permite incrementar la promoción de nuestra aplicación, haciendo que sea conocida por muchas mas personas.

En conclusión todos los objetivos requeridos fueron cumplidos.

6.2 PASOS FUTUROS

Para empezar habría que completar nuestra aplicación con la funcionalidad de dejar comentarios en cada foto. De esta manera los usuarios podrían discutir sobre los fotos, seria una manera de comunicación entre ellos. Esta funcionalidad es muy útil ya que casi en todas las aplicaciones fotográficas de AppStore es posible dejar comentarios y se puede observar que a los usuarios tienden a dejar sus opiniones sobre las fotos.

También seria útil tener implementada la funcionalidad para aplicar filtros, efectos o marcos a los fotos para que la foto quede a tu gusto.

Muy interesante sería conseguir mas usuarios registrados. Hasta ahora hemos usado solo un bono para promocionar y como el resultado hemos recibido mas de mil visitas en la pagina de landing de la aplicación. Se podría invertir mas recursos monetarios en Google Ads y en otras alternativas de publicidad como redes sociales y blogs.

Actualmente estamos pensando en desarrollar una aplicación basada en Picthru. A diferencia de Picthru, en cual se juega con todas las fotos que hacen los usuarios con su Smartphone, en esta nueva aplicación se jugará con las fotos que los usuarios tienen en el Facebook.

7. MANUAL DE ADMINSTRADOR

Manual de instalación de servidor:
El sistema operativo puede ser cualquiera, solo si es posible realizar siguientes instalaciones:

1. Instalar Apache,PHP,MySql,PhpAdmin, CakePHP:
   http://www.maestrosdelweb.com/editorial/phpmysqlap/
2. Copiar la carpeta de cakepicthru2(de cakephp) en la carpeta htdocs de apache
3. Crear BD con el nombre Picthru2 y insertar en ella las siguientes tablas:

```
CREATE TABLE `cake_sessions` (
`id` varchar(255) collate latin1_german2_ci NOT NULL default '',
`data` text collate latin1_german2_ci,
`expires` int(11) default NULL,
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_german2_ci;

CREATE TABLE `combos` (
`id` int(10) NOT NULL auto_increment,
`parent1_id` int(10) unsigned NOT NULL default '0',
`parent2_id` int(10) unsigned NOT NULL default '0',
`user_id` int(10) default NULL,
`created` datetime NOT NULL default '0000-00-00 00:00:00',
`updated` datetime NOT NULL default '0000-00-00 00:00:00',
`description` varchar(140) character set latin1 NOT NULL default '',
PRIMARY KEY  (`id`),
UNIQUE KEY `thru1_id` (`parent1_id`,`parent2_id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_german2_ci
AUTO_INCREMENT=51 ;

CREATE TABLE `follows` (
`id` int(10) NOT NULL auto_increment,
`id_user_follower` int(10) NOT NULL default '0',
`id_user_following` int(10) NOT NULL default '0',
`created` datetime NOT NULL default '0000-00-00 00:00:00',
PRIMARY KEY  (`id`),
UNIQUE KEY `id_user_follower` (`id_user_follower`,`id_user_following`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 COLLATE=latin1_german2_ci
AUTO_INCREMENT=55 ;

CREATE TABLE `infopoints` (
`id` int(11) NOT NULL auto_increment,
`player_id` int(11) NOT NULL default '0',
`created` datetime NOT NULL default '0000-00-00 00:00:00',
`type` int(3) NOT NULL default '0',
`points` int(11) default '0',
`pointsbefore` int(11) NOT NULL default '0',
`penaltypoints` int(11) NOT NULL default '0',
`pic_id` int(11) default NULL,
```

```
`from_player_id` int(11) default NULL,
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=376 ;

CREATE TABLE `infosets` (
`id` int(11) NOT NULL default '1',
`level` int(11) NOT NULL default '1',
`date_modified` date NOT NULL default '0000-00-00',
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

CREATE TABLE `pics` (
`id` int(10) unsigned NOT NULL auto_increment,
`player_id` int(10) unsigned default NULL,
`combo_id` int(10) unsigned NOT NULL default '0',
`parent1_id` int(10) unsigned default NULL,
`parent2_id` int(10) unsigned default NULL,
`playerPhotoFile` varchar(50) default NULL,
`playerPhotoLikes` int(10) unsigned NOT NULL default '0',
`playerPhotoNotLikes` int(10) unsigned NOT NULL default '0',
`created` datetime default NULL,
`description` varchar(140) default NULL,
`picthru` int(1) NOT NULL default '0',
`timeschoosen` int(10) NOT NULL default '0',
`level` int(10) default NULL,
`indexInlevel` int(10) default NULL,
`choosingIndexInLevel` int(10) NOT NULL default '0',
`community` tinyint(1) NOT NULL default '1',
`rep_num` int(6) NOT NULL default '0',
`index_rep` int(6) NOT NULL default '0',
`num_tries` int(11) NOT NULL default '0',
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=1162 ;

CREATE TABLE `userlevels` (
`id` int(11) NOT NULL default '0',
`set1` int(11) NOT NULL default '0',
`set2` int(11) NOT NULL default '0',
`set3` int(11) NOT NULL default '0',
`set4` int(11) NOT NULL default '0',
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1;

CREATE TABLE `users` (
`id` int(11) NOT NULL auto_increment,
`username` varchar(50) default NULL,
`password` varchar(40) default NULL,
`score` int(11) NOT NULL default '0',
`email` varchar(50) NOT NULL default '',
`profilePhoto` varchar(50) default NULL,
```

```
`level` int(10) NOT NULL default '1',
`level_community` int(10) NOT NULL default '1',
`set_community` int(11) NOT NULL default '1',
`last_played_level_set1` int(11) NOT NULL default '1',
`nrep_level_comunity` int(10) NOT NULL default '0',
`num_pic_level_community` int(10) NOT NULL default '0',
`created` datetime default NULL,
`ip` varchar(100) default NULL,
`country` varchar(200) default NULL,
`date_last_login` datetime default NULL,
`date_last_logout` datetime default NULL,
`notifToken` varchar(100) default NULL,
PRIMARY KEY  (`id`),
UNIQUE KEY `username` (`username`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=175 ;


CREATE TABLE `variables` (
`id` int(3) NOT NULL auto_increment,
`current_community_level` int(10) default '1',
PRIMARY KEY  (`id`)
) ENGINE=MyISAM DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;
```

Login

Para crear nuevos niveles en cada set el admin debe logearse en esta dirección http://app.picthru.net/cakepicthru2/users/login. La pantalla para login en el pagina web de la aplicación es la imagen 7.

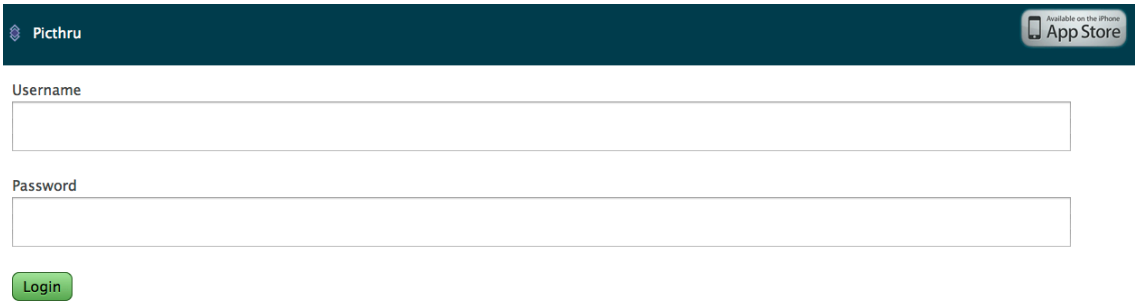


Imagen 7

Si el login se realizo correctamente aparecerá la pantalla que se puede observar en la imagen 8. En esta pantalla aparecen los cuatro sets y en cada set se ven los imágenes con las que juegan los usuarios en cada nivel.

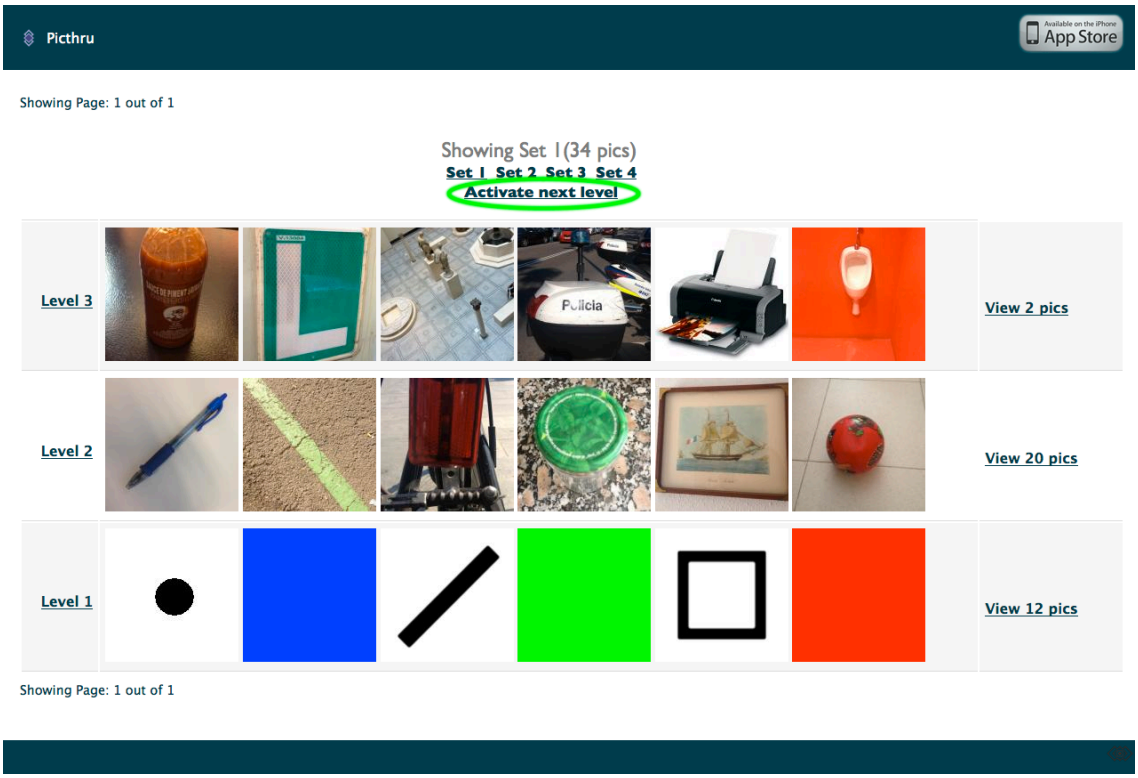


Imagen 8

Consulta de las imágenes de nivel

Podemos ver cuantas imágenes han hecho en cada nivel y toda la información relativa a ellas como se puede ver en la imagen 9.

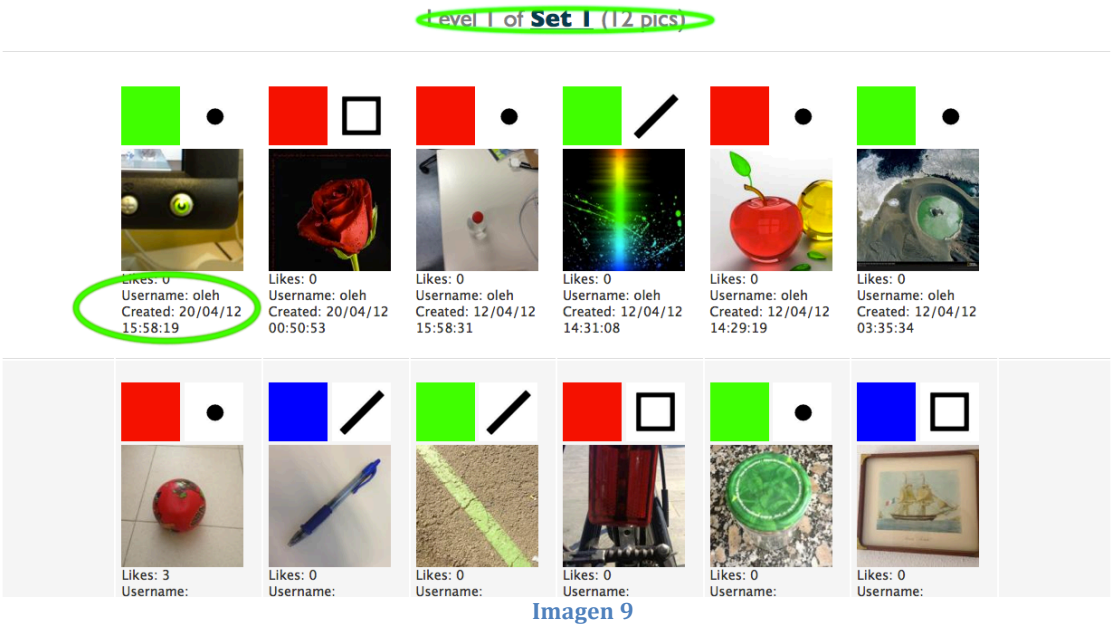


Imagen 9

Activación de nivel

Para activar el siguiente nivel para un set, pulsamos el enlace correspondiente. Nos aparece la pantalla que se puede ver en la imagen 10.

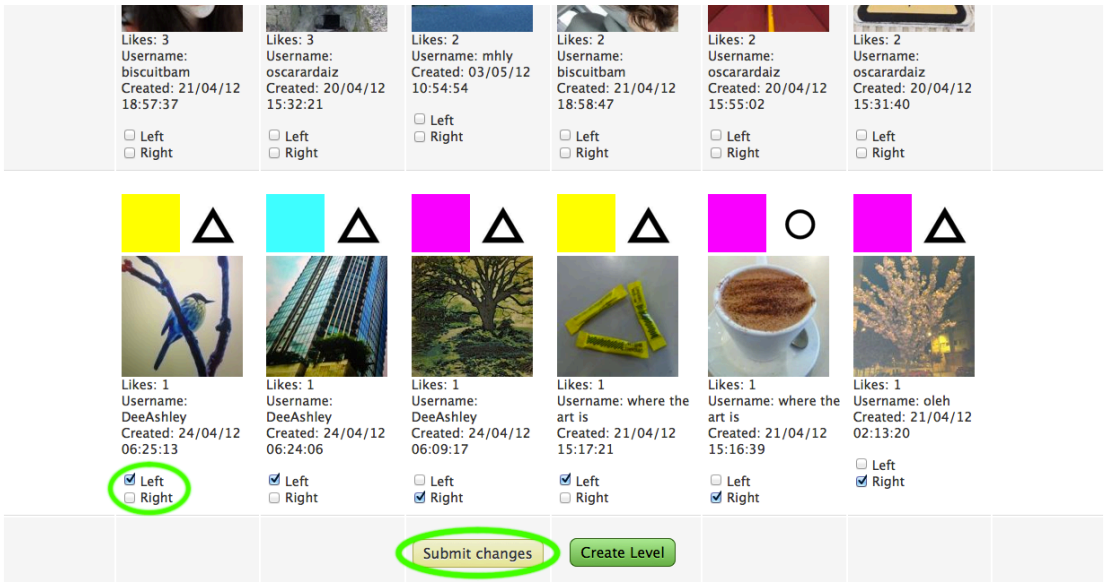


Imagen 10

Elegimos las tres imágenes que queremos para el lado izquierdo y las tres que queremos para el lado derecho y pulsamos el botón para efectuar cambios. Como resultado obtenemos la pantalla de la imagen 11 en la cual ya podemos activar el siguiente nivel pulsando sobre el botón seleccionado con un circulo.

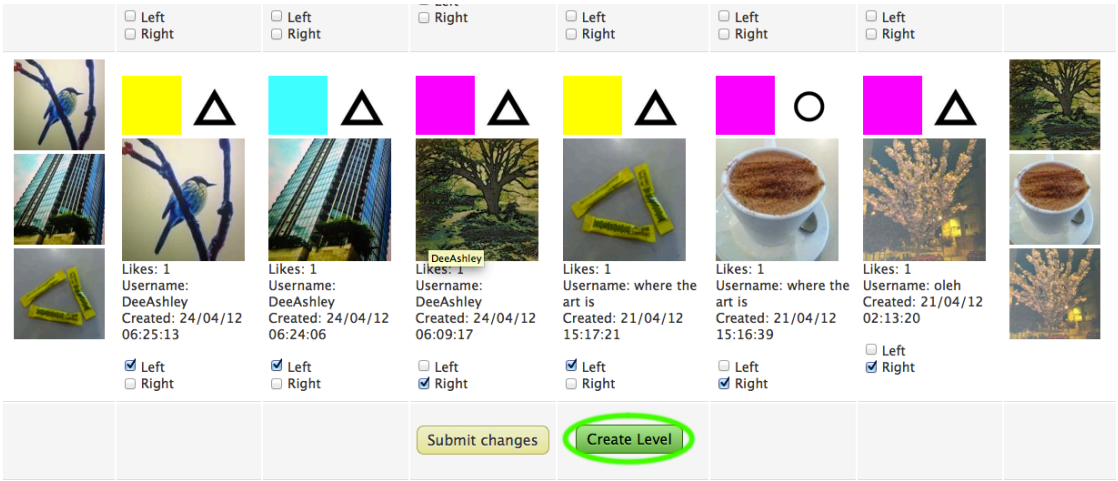


Imagen 11

8. MANUAL DE USUARIO

Usuarios no registrados

Los usuarios no registrados solo pueden ver la pantalla que permite seleccionar la combinación de dos imágenes y la pantalla donde se puede ver las imágenes que han hechos los usuarios registrados con las combinaciones seleccionadas. Son las pantallas de las imágenes 12 y 13 respectivamente.

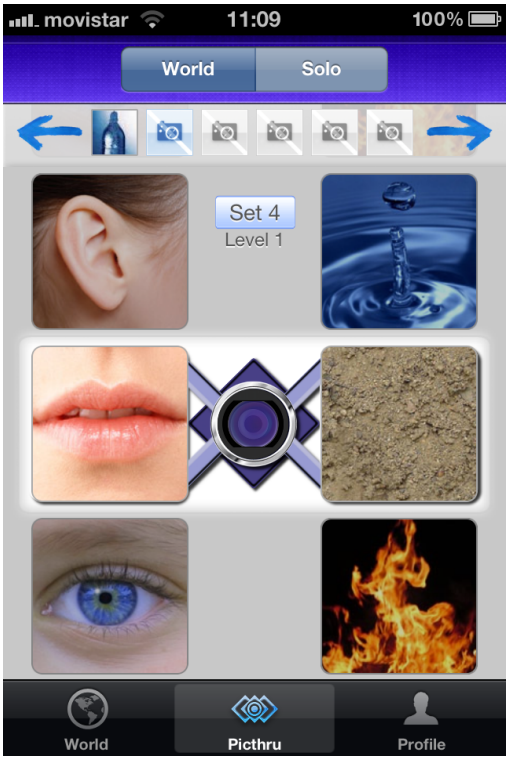


Imagen 12

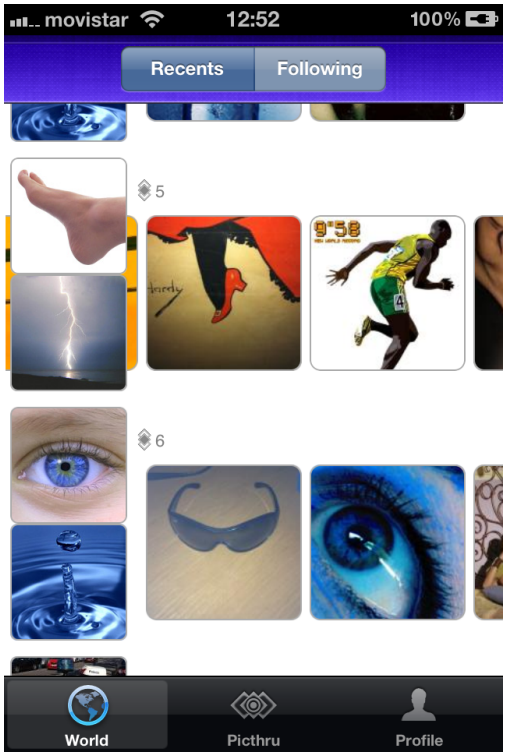


Imagen 13

Usuarios registrados

Sign Up y Login

Para poder sacar fotos y ver todas las pantallas de la aplicación los usuarios tienen que registrarse. La pantalla para registrarse aparece cuando el usuario pulsa sobre la cámara en la pantalla de la imagen 12 o cuando el usuario quiere acceder a cualquier otra pantalla de la aplicación. La pantallas de Sign Up y Login se puede ver en las imágenes 14 y 15 respectivamente.

Imagen 14

Imagen 15

Elección de la combinación de dos imágenes.

Podemos elegir diferentes conjuntos de imágenes con las que queremos hacer combinaciones. Para cambiar los conjuntos hay que pulsar las flechas en la pantalla de la imagen 16. Por ejemplo en la pantalla de la imagen 17 se puede observar otro conjunto de la imágenes.

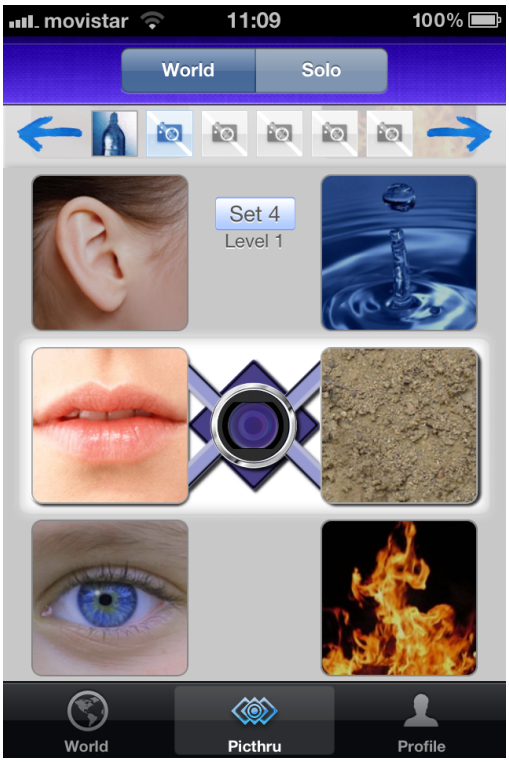


Imagen 16

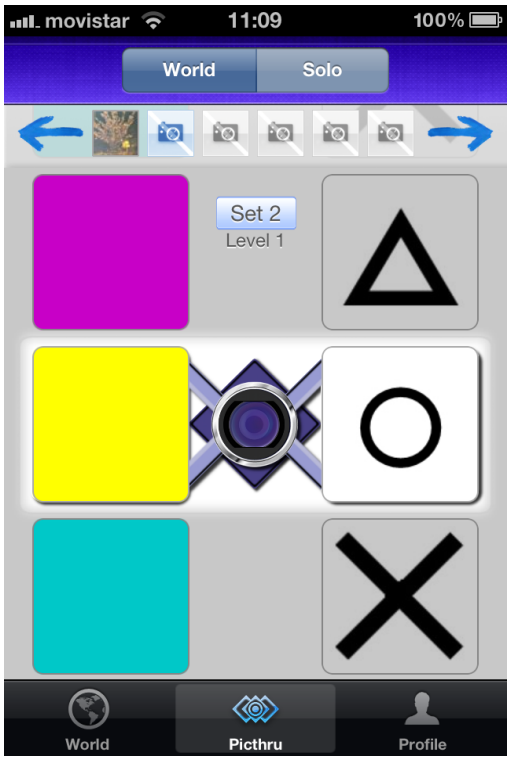


Imagen 17

Después de haber elegido el conjunto de imágenes con las que vamos a hacer combinaciones, el siguiente paso sería escoger dos imágenes. En la pantalla de la imagen 18 se puede observar como se elige aleatoriamente una combinación. Las imágenes de la izquierda y de la derecha empiezan a moverse cuando el usuario las toca o intenta mover, y después de unos segundos se detienen y aparece una cámara, como en la pantalla de la imagen 19, para sacar una foto. Si el usuario no saca una foto para la combinación elegida, esto se contará como un fallo.

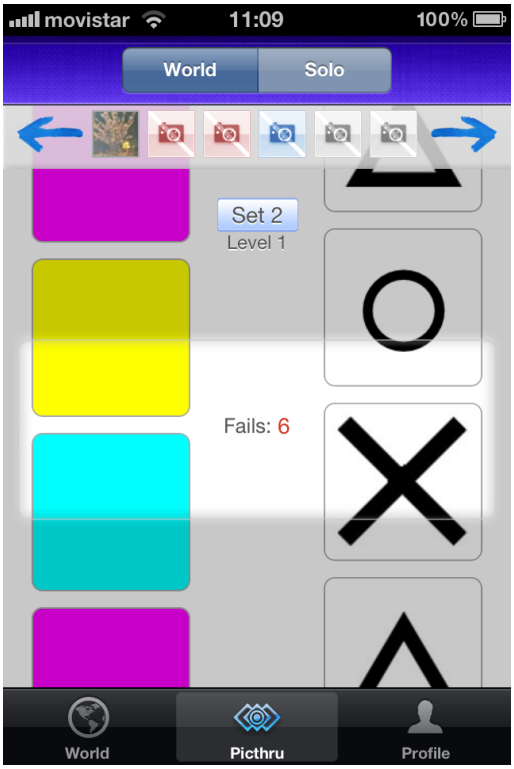


Imagen 18

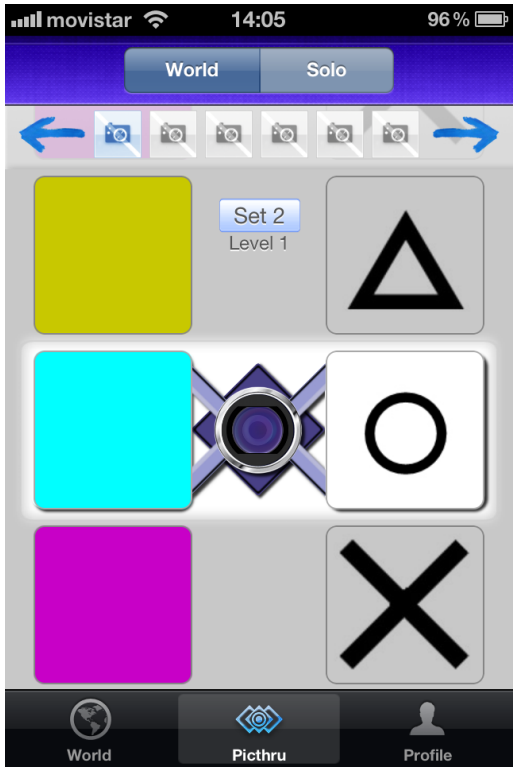


Imagen 19

Como sacar una foto para una combinación

Cuando pulsamos el botón de la cámara en la pantalla de la imagen 19 se abre la pantalla de la cámara que se puede ver en la imagen 20. Podemos hacer una foto directamente con la cámara o elegir una foto del álbum. En la pantalla de la imagen 21 la foto fue hecha con la cámara.

Imagen 20

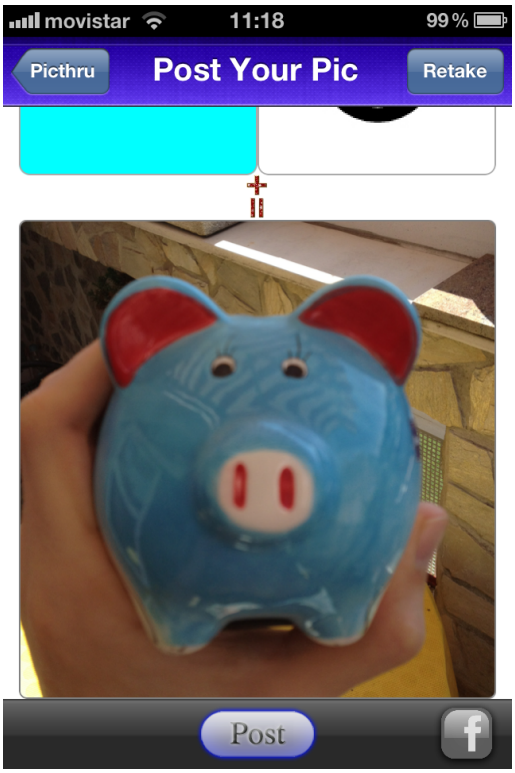


Imagen 21

En las pantallas de la imágenes 22 y 23 la foto se elige desde el álbum. También es posible recortar la foto ya que todas las fotos de la aplicación deben ser cuadradas.

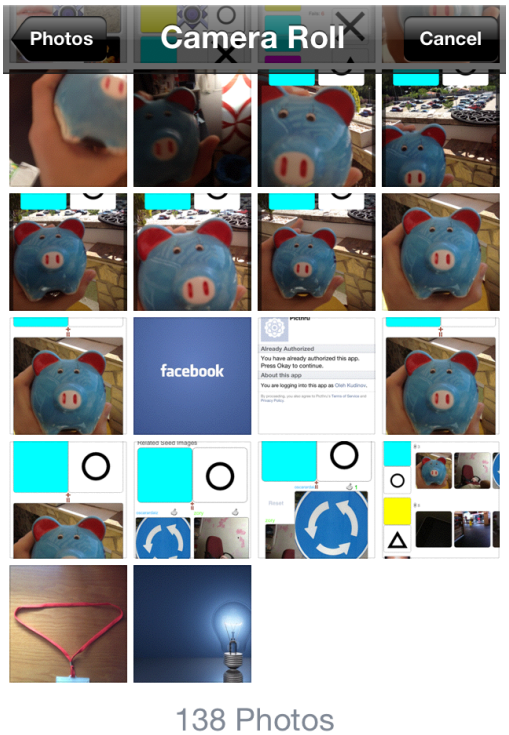


Imagen 22

Imagen 23

Subir foto al servidor de la aplicación y a Facebook

Para subir una foto a Facebook hay que activar el botón con la letra f y pulsar el botón de Post en la pantalla de la imagen 24. La foto se subirá al servidor de Picthru y también a Facebook. También se podrá escribir comentarios para la foto que se sube a Facebook como se puede observar en la pantalla de la imagen 25.

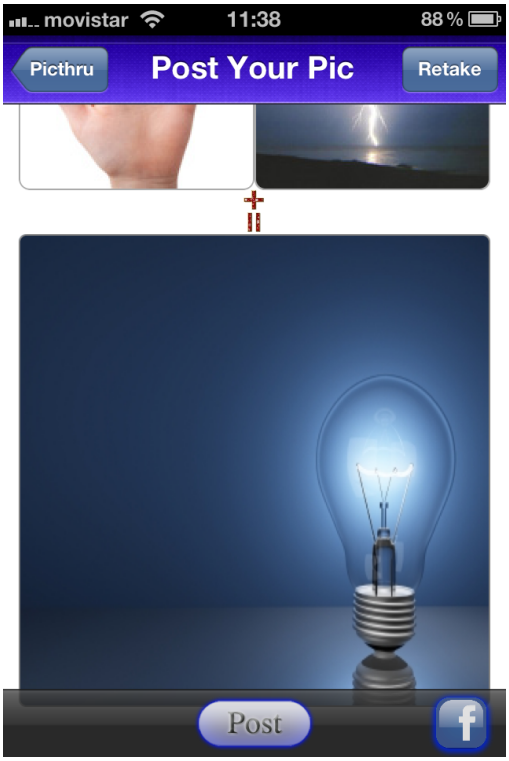


Imagen 24

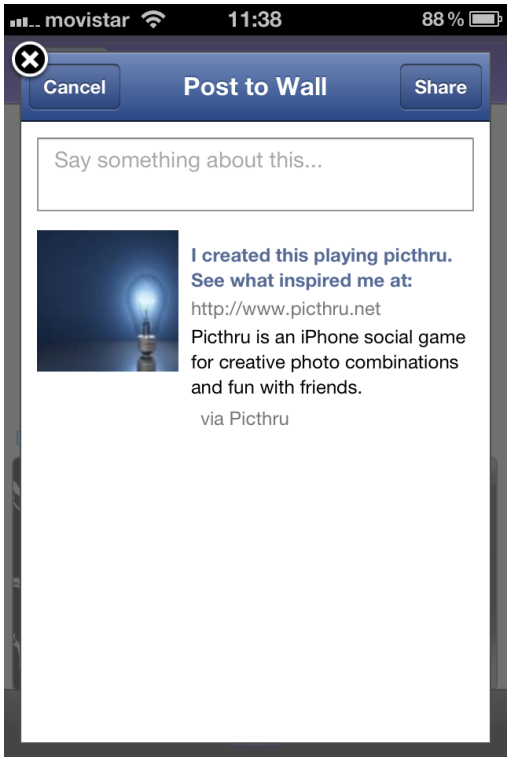


Imagen 25

La foto que se sube al servidor de la aplicación aparecerá en la pantalla de la imagen 26 y en la pantalla de perfil de la imagen 27. Todos los demás usuarios podrán ver estas fotos y dar sus votos por ellas.

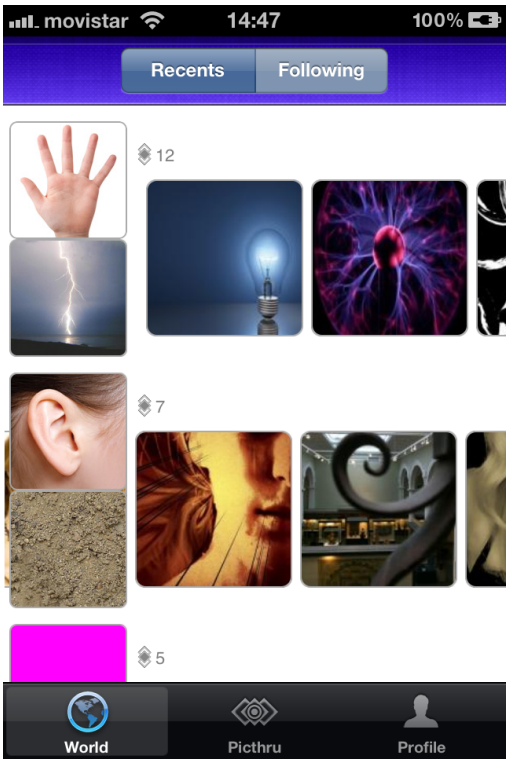


Imagen 26

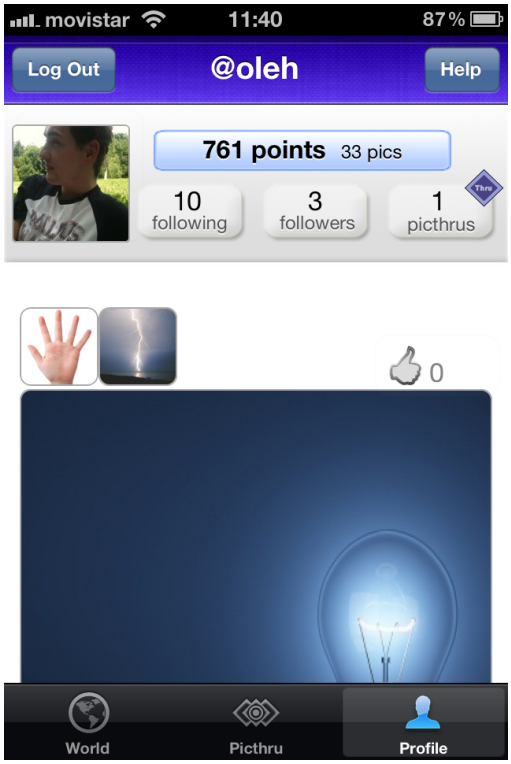


Imagen 27

Votación por las imágenes

Cuando hacemos una imagen para una combinación y la subimos al servidor nos aparece la pantalla de la imágenes 28 y 29 en la cual tenemos que elegir la mejor imagen para esta combinación.

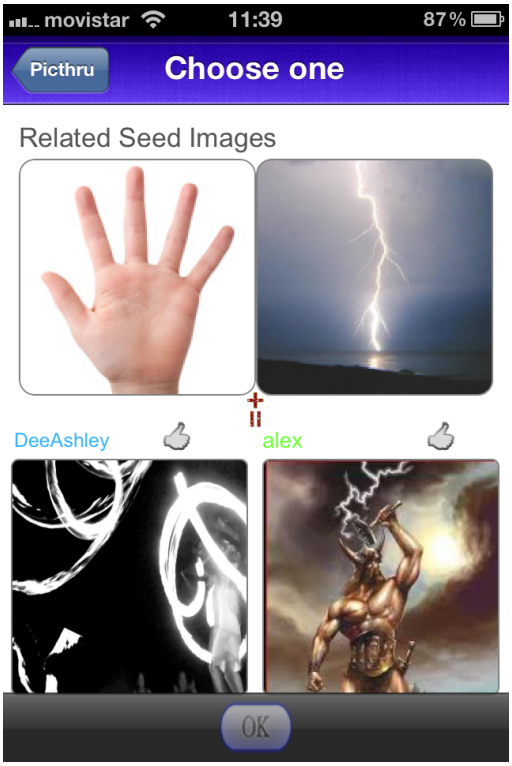


Imagen 28

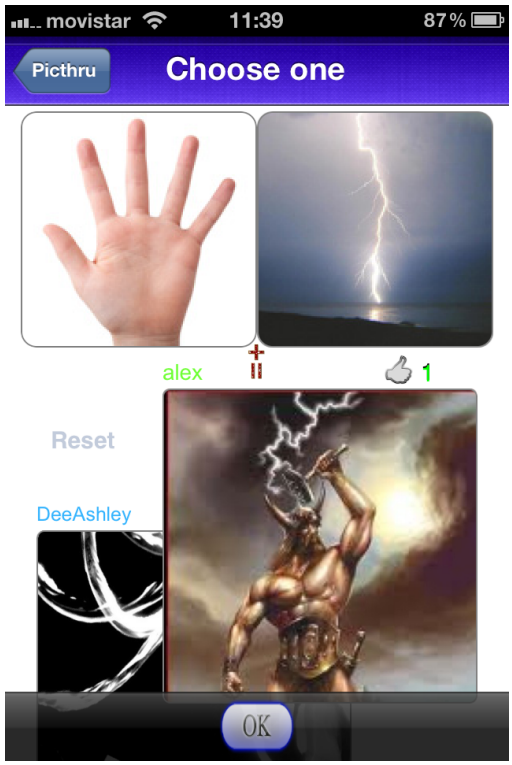


Imagen 29

Visualización del árbol de fotos

Las fotos que se suben a Facebook tienen un link al sitio web de la aplicación donde se puede ver el árbol de todas las combinaciones previas a la foto que hemos subido. La pantalla de este árbol se puede ver en la imagen 30.

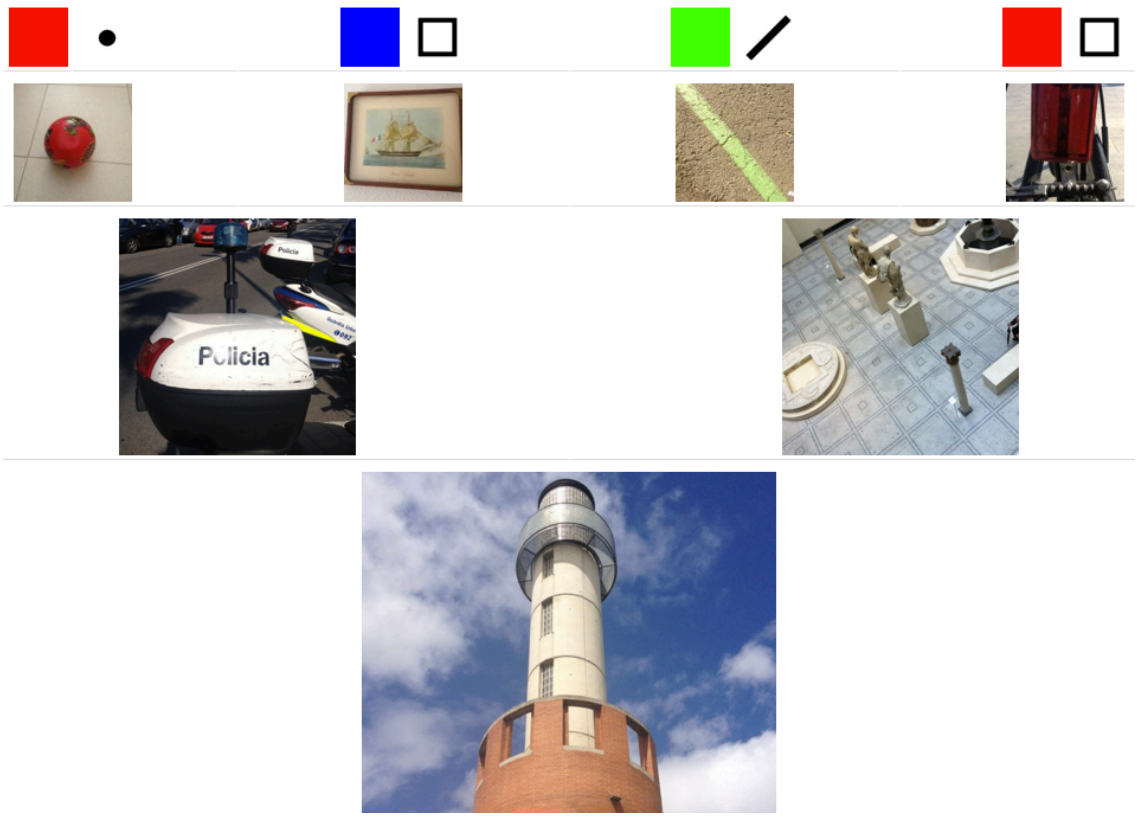


Imagen 30

También se puede ver el árbol parcial desde el IPhone. Cuando pulsamos en una imagen cualquiera la pantalla de la imagen 31 se recarga y enseña la combinación a la que pertenece esta imagen, por ejemplo si pulsamos sobre el balón el resultado es la imagen 32.

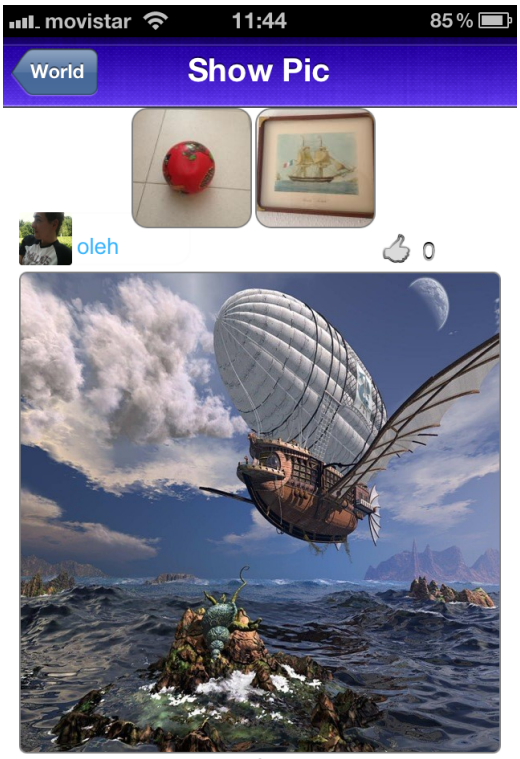


Imagen 31

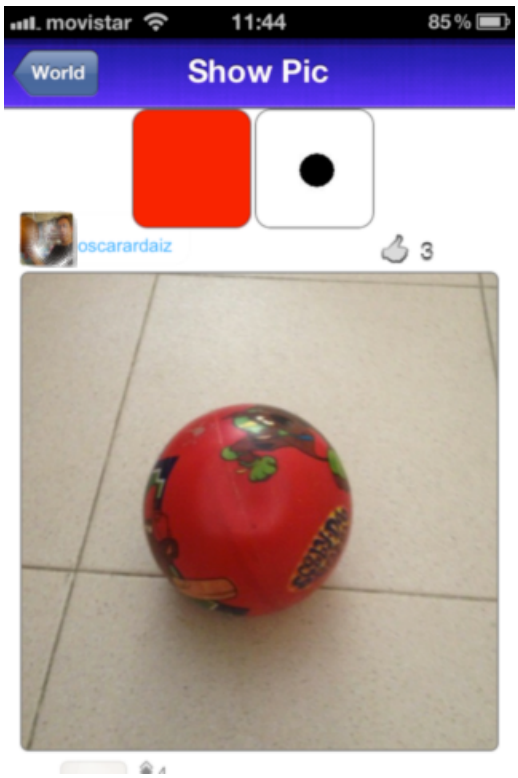


Imagen 32

Si bajamos el scroll de la pantalla de la imagen 30 podríamos ver todas las combinaciones existentes que incluyen el balón. Esta pantalla esta en la imágenes 33 y 34.

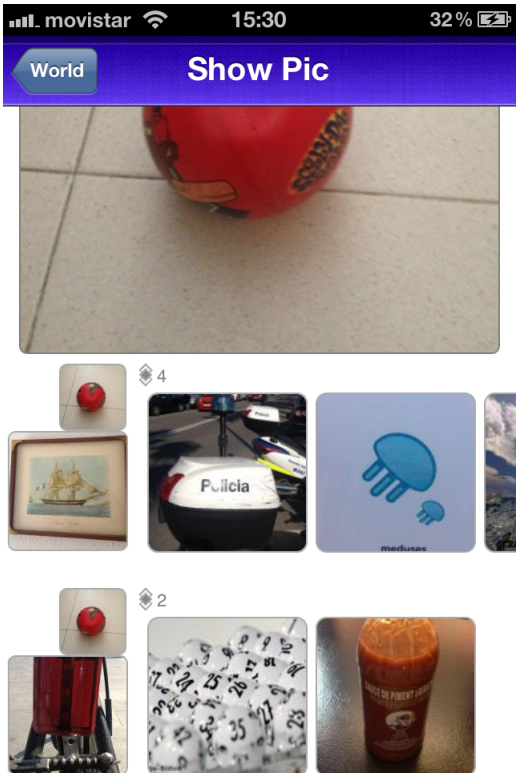


Imagen 33

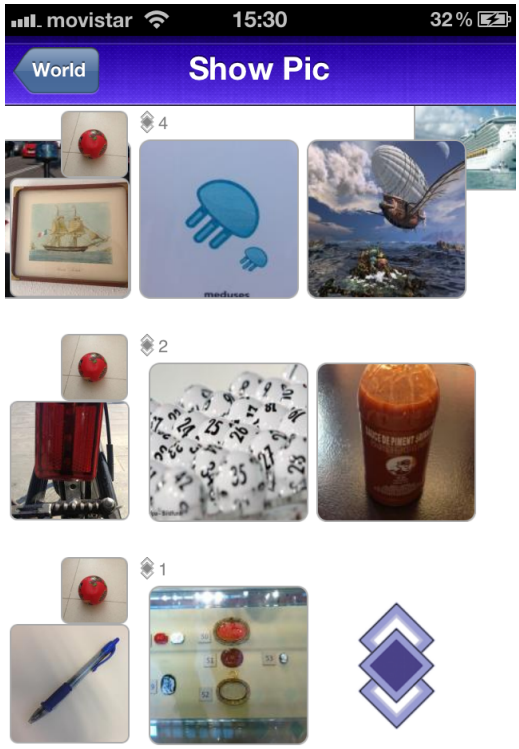


Imagen 34

Notificaciones y actividad de los usuarios

Las notificaciones avisan mediante un mensaje a los usuarios sobre una actividad realizada. Por ejemplo en caso de que a alguien le guste mi foto o alguien me empiece a seguir me llegará una notificación que se podrá observar en la esquina superior de la imagen 35. Después de entrar en la aplicación veríamos un circulo en rojo que se puede ver en la pantalla de la imagen 36. Cuando el usuario pulsa sobre el botón con el circulo rojo se le abrirá la pantalla de la actividad de la imagen 37. En esta pantalla se podrá ver como el usuario ha conseguido los puntos y la actividad de los usuarios.

Imagen 35

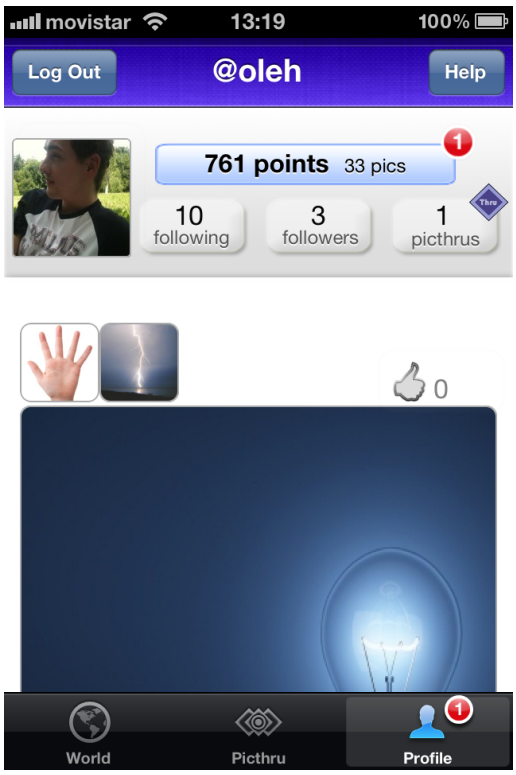


Imagen 36

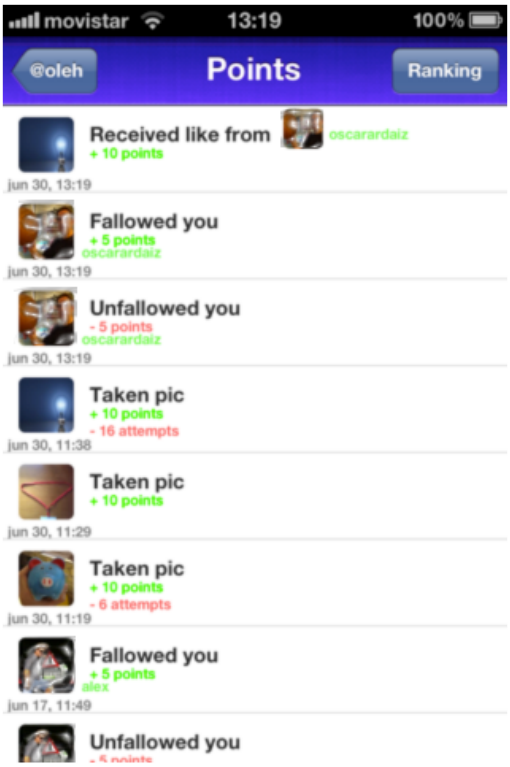


Imagen 37

Seguir a los usuarios y ver a los seguidores.

Para seguir a un usuario tenemos que entrar en su pantalla de perfil y pulsar el botón de seguir que esta en la esquina superior en la imagen 38.
Los seguidores se puede ver desde el perfil. Esta pantalla esta en imagen 39.

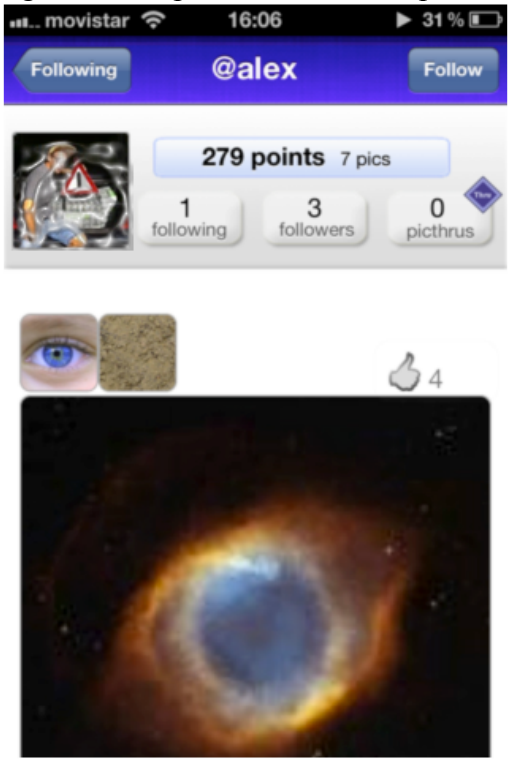


Imagen 38

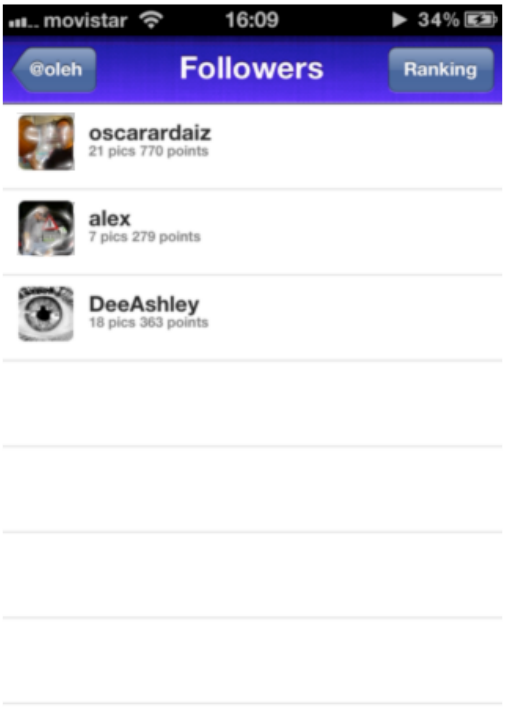


Imagen 39

Ver picthrus de usuarios.

Como ya sabemos, pichtru es una foto que ha sido escogida como mejor foto para una combinación y con esta foto los usuarios pueden formar nuevas combinaciones y jugar con ellas. Esta pantalla se abre desde el perfil. En la imagen 40 se puede ver esta pantalla. Las picthrus llevan encima un rombo azul con la palabra Trhu.

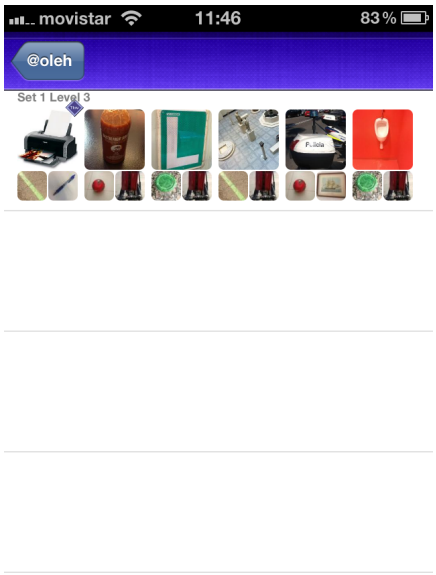


Imagen 40

Pantalla de ayuda

La pantalla de ayuda se puede ver en la imagen 41. Se accede desde el perfil y la pantalla de login.

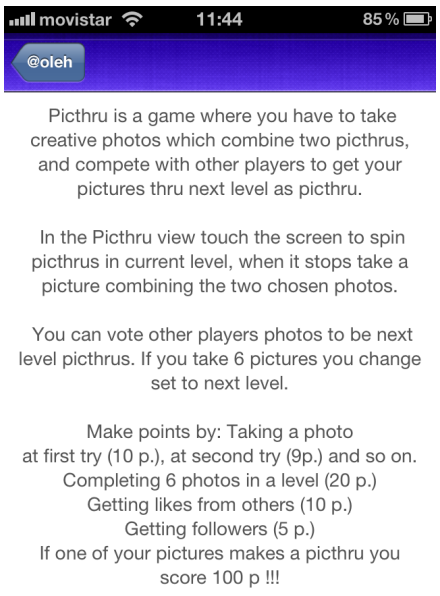


Imagen 41

Ver todos los niveles de un conjunto de fotos picthtrus

Para ver la pantalla de todos los niveles de un conjunto hay que pulsar sobre el botón Set en la pantalla de la imagen 42. Se abrirá esa pantalla que se puede observar en la imagen 43.

Imagen 42

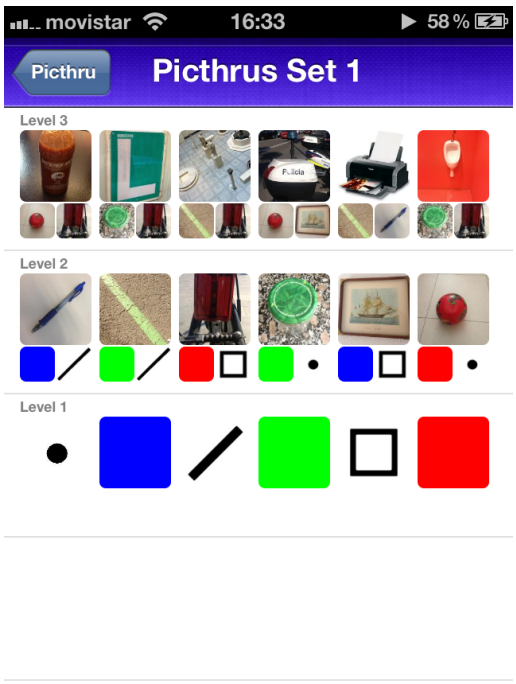


Imagen 43

9. EQUIPOS UTILIZADOS EN EL PFC

El desarrollo de la aplicación se ha realizado en el equipo Mac mini de la universidad. Las especificaciones de este equipo son:

- Procesador
  - Procesador Core i5 de Intel de doble núcleo a 2,3 GHz con 3 MB de caché integrada de nivel 3 compartida
- Memoria
  - 2 GB de memoria DDR3 a 1.333 MHz
  - Disco duro de 500 GB[2] a 5.400 rpm
- Gráficos
  - Procesador HD Graphics 3000 de Intel con 288 MB de SDRAM DDR3 compartida con la memoria principal[3]
- Software incluido:
  - Sistema Operativo X Lion
  - Xcode

Para la compilación y las pruebas de la aplicación se ha usado el IPhone 4S. Las especificaciones técnicas de este Smartphone son:

- Procesador
  - Chip A5 de Apple de doble núcleo a 1 GHz
- Memoria
  - Memoria RAM: 512 MB eDRAM
  - Memoria flash de 16 GB
- Gráficos
  - Integrados en el chip A5
- Software incluido:
  - Sistema iOS 5.1
- Redes móviles e inalámbricas
  - Wi-Fi 802.11b/g/n (802.11n solo a 2,4 GHz)

10. REFERENCIAS

[1] A. Kultima, J. Niemelä, J. Paavilainen, and H. Saarenpää, Designing game idea generation games. In Proc. Of Future Play '08. ACM Press, 137-144. 2008.

[2] B. Shneiderman. Creativity support tools: Report from NSF sponsored workshop. Int. J. Human-Computer Interaction 20 (2), 61-77. (2006).

[3] L. Yu, and J.V. Nickerson, Cooks or Cobblers? Crowd Creativity through Combination, CHI 2011, ACM Press (2011).

11. BIBLIOGRAFÍA

- Librería de desarrollador iOS(sistema operativo de IPhone)
https://developer.apple.com/library/ios/navigation/

- Integración de iOS con plataforma de Facebook .
http://developers.facebook.com/docs/mobile/ios/build/

- Integración de Push Notificaciones con iOS:
https://docs.urbanairship.com/display/DOCS/Getting+Started

- Manual de CakePHP:
http://book.cakephp.org/1.3/

- Framework para iOS para interacción con servicios web de RESTful
http://restkit.org/

- Todo sobre PHP:
http://www.desarrolloweb.com/php/

- Información sobre Piictu:
http://appleweblog.com/2011/09/piictu-para-ios-la-fotografia-hecha-conversacion

- Informacion sobre aplicaciones fotográficas para Smartphone:
http://www.xatakafoto.com/guias/las-10-mejores-aplicaciones-fotograficas-para-ios
http://smartphonesworld.es/aplicaciones-fotograficas-1/

- Wikipedia
http://es.wikipedia.org/wiki/Apple_Push_Notification_Service
http://es.wikipedia.org/wiki/Xcode

- Fugu FTP
http://codigosur.org/article/fugu-ftp-ftp-p-mac/